MIKROSZÁMÍTÓGÉP
MAGAZIN
Ára: 30 Ft
1986
április
A NEUMANN JÁNOS SZÁMÍTÓGÉPTUDOMÁNYI TÁRSASÁG LAPJA
2c

JÓ DÖNTÉS!

KEDVEZŐ FELTÉTELEKKEL JUT SZEMÉLYI SZÁMÍTÓGÉPES HÁLÓZATHOZ,

ha

PROPER—16

gépet vásárol

AZ SZKI—SCI—L BŐVEBB PROPER VÁLASZTÉKOT KÍNÁL: KORÁBBAN AZ SZKI-TÓL VÁSÁROLT GÉPEIBŐL SZEMÉLYI SZÁMÍTÓGÉPES HÁLÓZATOT ALAKÍTHAT KI!

ÚJDONSÁG

— többfelhasználós rendszerek
— PROPER—16-os gépekből kialakított helyi hálózat
— új mega PC: maximálisa memóriával (704 kbájt) színes monitorral, gyors háttértármentéssel 1 millió forint alatt
— világszínvonalú perifériák

RENDKÍVÜL JÓ ÁR/TELJESÍTMÉNY VISZONY!

EZ A MA TECHNIKÁJA

Szki

INFORMÁCIÓ:
Budapest I., Iskola u. 10.
Telefon: 350-180
1011

SZÁMÍTÁSTECHNIKAI KUTATÓ INTÉZET ÉS INNOVÁCIÓS KÖZPONT
Budapest I., Donáti u. 35—45.
1015

Sci-L

Scitel

MIKROSZÁMÍTÓGÉP MAGAZIN

4. ÉVFOLYAM
3. SZÁM
1986. ÁPRILIS

A NEUMANN JÁNOS SZÁMÍTÓGÉP-TUDOMÁNYI TÁRSASÁG LAPJA

A kiadvány a Tudományszervezési és Informatikai Intézettel együttműködve készül

A szerkesztőbizottság vezetője:
Kovács Győző

E számunkat szerkesztették:

Bakos Tamás (programozástechnika)
Broczkó Péter (hírek)
Kovács Győző (levelezés)
Lindner László (sakkprogramozás)
Petróczy Judit (könyvek)
Pogány Csaba (alkalmazástechnika, tanfolyam)
Simonyi Endre (klub)
Takácsy Ildikó (favágás)
Vadkerti János (μprogramok)
Varga András (iskola — számítógép)

A szerkesztőség munkatársa:
Kardos Zsuzsa

Felelős szerkesztő:
Könyves Tóth Pál
Szerkesztőség:
1027 Budapest II., Fő u. 68.
Telefon: 154-250

Kiadja a Delta Szaklapkiadó és Műszaki Szolgáltató Leányvállalat
Felelős kiadó:
Faklen Pál igazgató
1142 Budapest VII., Garay u. 5.
Telefon: 415-583, 215-440

Terjeszti a Magyar Posta
Előfizethető bármely postahivatalban, a kézbesítőknél, a Posta hírlapüzleteiben és a Hírlapelőfizetési és Lapellátási Irodánál (Budapest V., József nádor tér 1. Postacím: 1900 Budapest) közvetlenül vagy postautalványon, valamint átutalással a HELIR 215—96162 pénzforgalmi jelzőszámra.
Megjelenik havonta
Példányonkénti ára 30,— Ft
Előfizetési díj:
egy évre 360,— Ft
fél évre 180,— Ft

Szikra Lapnyomda
Budapest (86-3080)
Felelős vezető:
Csöndes Zoltán vezérigazgató

INDEX: 25629
ISSN 0236-6088

Címképünk:
A Commodore PC 20. Kapható a Novotrade RT. 2C Áruházában

Tartalom

Informatikai nyelvújítás 2
A jó vezetés titka 12
Gyors importra van szükség! 13
A minőségügy közügy 17
A 65XX mikroprocesszor gépi és assembler szintű utasításkészlete 23
Assemblerek, cross-assemblerek 34
VC20-ak egymás között 38
Adok — veszek — cserélek 39
Hány sima, hány fordított? 40

ISKOLA — SZÁMÍTÓGÉP

Pályázat közoktatási számítógépes programok elkészítésére 3
Gép a gépen II. 4
Számítástechnikai tanfolyam egy amerikai iskolában 5
Egy tisztán szoftver úton megvalósított beszédszintetizátor 7
Káosz a Spectrumban 8

TANFOLYAM

Alapozás XVI. 9

PROGRAMOZÁSTECHNIKA

Strukturált programtervezés 14

μPROGRAMOK 20

μKLUB

PTA—4000 27
SPECTLOADER 28

DIÁKROVAT

Karaktergrafika normál üzemmódban 30
Képszerkesztő 31
Apró fogás 31
A számítógép anatómiája 32
Billentyűzetfeljavítás 33

FÓRUM 41

AZ OLVASÓ ÍRJA 42

FAVÁGÁS 44

SAKKPROGRAMOZÁS

A számítógépek különleges képességei 45
Bitek és figurák 46

KÖNYVEK 47

HÍREK, ÉRDEKESSÉGEK 48

Informatikai nyelvújítás

„Az ortographia fatalis dolog a magyarba; a Cancellarián alig van néhány, aki írni tud; minden magának külön csinál; ha tudsz te néhány olyas szók írására nézve s érkezel, tudósíts; mert te már sokat nyomtattattál."

(BOLYAI FARKAS LEVELE BODOR PÁLHOZ. MAROSVÁSÁRHELY, 1817. JANUÁR 4.)

Vannak témák, amelyeket gyakran elő kell venni, különösen, ha erre még a figyelmét is felhívják az embernek. Az informatikai szaknyelvről egyszer már írtam a µM 1985/1. számában *Az informatika társadalmasítása avagy a társadalom informatizálása* címmel. A Népszabadság 1985. december 24-i számában Boldizsár Iván „nyelvérzéki indulata" tiltakozik a *Naplócska — Kemény és lágy* című cikkében az informatika két „szörnyszülöttjének", a hardver és szoftver szavaknak a használata ellen — mint már olyan sokszor leírtam —, jogosan. Persze vannak ennél szebb példák is, mint a fájl (file), a ticsver (teachware), de nem rossz a förmver (firmware) sem.

Használunk eredeti magyar szószörnyeket is, amelyek közül hirtelen kettő jut eszembe. Mind a kettő betűszóként kezdte pályafutását. Az egyik a NYÁK (NYomtatott ÁramKör), amelyet már kisbetűvel írva is láttam, de a másik még szebb: BOÁK (Berendezés Orientált ÁramKör). Ez kisbetűvel írva, ragozottan a legszebb. Csak emlékezetből idézem a példát: „a boákokat először be kell mérni . . .". Ennél talán még a szoftver is elfogadhatóbb!

Nem szeretjük a fonetikusan leírt angol szavakat, mégis használjuk. Ha nem szólnak érte, akkor az eredeti angol alakban írjuk, ha szólnak, akkor bizony fonetikusan. Ez persze nem jelenti azt, hogy nem volna jobb, ha lennének jó és találó magyar szakkifejezések, de nagyon sok esetben nincsenek. Ha egy jó angol fogalmat egy kerek magyar mondattal lehet csak leírni, akkor már jobb az idegen szó; szinte olvashatatlan egy olyan írás, amelyben a lényeges szakkifejezéseket mindig meg kell magyarázni.

Sajnos az is előfordul, hogy van valamire találó magyar szakkifejezés, és mégsem használjuk. Példa erre éppen a számítógép sikerszavunk is, amelyet a szakma — néhány matematikus kivételével, akik következetesen számológépet használnak — szinte teljes egészében elfogadott, de a nem szakemberek, az írók és újságírók mégis a computert vagy a komputert használják, valószínűleg azért, hogy írásaikat színesítsék, vagy a szóismétléseket elkerüljék. De azt sem tartom lehetetlennek, hogy snobságból. Azt hiszik talán, hogy az angol computer szó tudományosabbá teszi írásukat.

Sokszor és sokan kísérleteztünk már, hogy olyan új kifejezéseket találjunk, amelyek Apáczai Csere Jánosnak a szavaival szólva: „újak, de a tudósok ítéletei szerint nem alábbvalók a régieknél". Számos tudós és nem tudós bizottság foglalkozik ma is a kérdéssel, mint például az Akadémia Nyelvtudományi Intézetében a Grétsy László vezette szaknyelvi bizottság vagy a Szabványügyi Hivatalban 1978 óta a számítástechnikai terminológiai bizottság, amely például kidolgozta és ki is adta a Számítástechnikai terminológiai szabványt. Meg kell említenem az NJSZT publikációs és terminológiai bizottságát is, amelynek legalább húsz éve egyik feladata a számítástechnikai nyelv idegen szakkifejezéseinek magyarítása. Több kiadvány is született, de még az 1973-ban kiadott Számítástechnikai Kislexikon sem tudott egyértelműen megbirkózni a feladattal. A Műszaki Könyvkiadó az előszóban a következőket írja:

„A közhasználatban az azonos jelentésű magyar és angol szavak keverednek. Sok esetben bele kellett törődnünk, hogy egy angol kifejezésre nem találunk használható vagy egyértelmű magyar fordítást. Ekkor az eredeti angol szót használtuk, általában az angol helyesírásnak megfelelő írásmóddal. Azonban közöltük azok kiejtés szerinti írásmódját is, hiszen hazánkban erre irányuló törekvések is vannak, és számos kiadványban ilyen formában találkozhat a szóval az olvasó."

Azóta nemcsak „erre irányuló törekvések" vannak, hanem idegen szavak esetében a kiejtés szerinti írásmód kötelező.

A negatív példák mellett vannak eredmények is, azaz olyan jó magyar kifejezések, amelyeket a szakmai nyelv gyorsan átvett és használ: tár (storage), adatátvitel vagy egy felesleges toldalékkal távadatátvitel (telecommunication), hálózat (network), billentyűzet (keyboard), bizonylat (document), cím (address), címke (label), csatorna (channel), futás vagy futtatás (run), cserélhető tár (removable storage), mágnesdob (drum storage), utasítás (instruction), elérési idő (access time), megszakítás (interrupt), fényceruza (light pen), egér (mouse), folyamatábra (flowchart), folyamatirányítás (process control), forrásnyelv (source language), rajzoló (plotter), hibajelzés (error indication), hívás (call), időosztás (time sharing), kapu (gate), kártya (card), fordítóprogram (compiler), lap (page), lemez (disk), nyomtató (printer), óra (timer), vagy távadatfeldolgozás (teleprocessing), hogy csak a leggyakoribbakat említsük.

Nem tudott viszont elterjedni néhány magyaros hangzású, sőt találó kifejezés mint például a bezúdítás (roll in), a morzsa (chip), a felülírás (overlay), a keverő rendezés (sort/merge), a (ki)bogarászás (debugging), a hajlékonylemez (floppy disk), a merevlemez (hard disk), a vezérmű (control unit), a töltés (loading) és mások.

Ezzel szemben vannak olyan kifejezések, amelyek nem túl szépek, nem is magyarosak, de azért szerintem érdemes használni őket írásban és szóban is: bevitel (input), kivitel vagy kiírás (output), beolvasás (read in), átlapolás (overlapping), az eljárásra orientált nyelv (procedure oriented language), a feltétlen vagy feltétel nélküli ugrási (ugró) utasítás (unconditional jump instruction), a képernyő (screen), a puffer (buffer) és még néhány hasonló.

Egyáltalán nincs jó kifejezésünk a már említett hardware és software mellett olyan szavakra sem, mint a byte, decode (dekódolni?), file, inverter, display, task, operation system, off-line, on-line, processor, routin, subroutin stb.

Végül vannak olyan elfogadott angol eredetű szavaink, amelyek helyett kár lenne mást keresnünk, hiszen a kiejtésük magyaros, és a fogalmat pontosan meghatározzák. Például: bit, maszk (mask), program, információ, informatika, mátrix, bináris (binary), decimális (decimal), alfanumerikus (alpha-numerical), analóg, karakter, szemantika (semantics), teszt (test), rekord (record), regiszter (register), kód (code), index, virtuális (virtual) stb.

Az elmúlt évek sikeres és sikertelen próbálkozásai azt mutatják, hogy nem könnyű olyan szavakat találni, amelyek a szóalkotás tartalmi és formai követelményeinek is megfelelnek. Szívesen ismételjük meg tehát a Boldizsár Iván cikkének végén elhangzott felhívást, és kíváncsian várjuk, hogy lesz-e kérésének új szavakban mutatkozó eredménye.

A felhívás: „Akit ez a kis nyelvújítás érdekel, kérem írja meg nekem, és majd megírom az újságban. Így talán sikerül meghonosítanunk az új szavakat".

Szeretnék még egy megjegyzést tenni. A számítógépek árának csökkenése azt eredményezte, hogy egyre többen kerültek közvetlen kapcsolatba az informatikával, és már nemcsak a munkaidejüket, de nagyon sokan a szabadidejüket is számítógépes munkával töltik. Saját becslésem szerint az informatika iránt érdeklődők száma nálunk is megközelíti már a milliót. Talán nem is nagy baj, ha ezeknek a valójában amatőröknek egy olyan magas szintű programnyelvet kell megtanulniuk, amelynek alaputasításai angol — esetleg torz angol — szavak. Nem egy tizenéves diákkal, de még kisebbel is találkoztam, aki azért kezdett el angolul tanulni, mert nemcsak szajkózni akarta a szavakat, hanem meg is akarta érteni azt, amit leírt. Ha nincs számítógép vagy számítástechnika, ezeknek a

gyerekeknek talán sohasem jut eszükbe, hogy idegen nyelvet tanuljanak. Így a programozási nyelv tanulásával nem csupán számítástechnikai, hanem humán ismereteiket is gyarapítják.

Azt hiszem, nem kétséges, hogy az informatikai nyelvújításra szükség van, a „fülsértő és szemrontó" kifejezéseket ki kell irtani az informatikai szaknyelvből.

Azt mondja az Új Magyar Lexikon: „... mozgalomszerű nyelvújításra akkor szokott sor kerülni, amikor — többnyire a nemzeti fejlődés velejárójaként vagy valamely idegen nyomásra, hatásra való visszahatásként — felvirágzik az irodalom, fellendülnek a tudományok, megerősödik a nyelvtisztítást sürgető nemzeti öntudat. A magyar nyelvújítás a 18. sz. utolsó és a 19. sz. első évtizedeiben — a nemzetté válás velejárójaként — nagy méreteket öltött, szinte minden tollforgató ember belekapcsolódott. Először a különféle szaktudományok kifejezéskészletének az idegen elemektől való megtisztítása volt a cél, ...".

A magyar irodalmi nyelvújítás célja az elavult szavak felújítása, új szóösszetételek és képzések alkotása mellett a tudomány fejlődésével meghonosodott idegen, főleg latin és görög, de más nyelvekből is ide származott szavak lefordítása volt.

Hogy e megvalósulás túl gyorsan nem sikerült, azt két, nem is akárkitől származó idézettel szeretném bemutatni. Az első egy levél, amelyet Apáczai Csere János írt Barcsai Ákosnak 1659. január 27-én:

„Apátzai János alázatos supplicatiója Nagyságodhoz, mint igen kegyelmes urához és Maecenássához.

Kegyelmes Uram! Alázatosan jelentem Nagyságodnak, hogy az a schola, amely az egész Keresztyénség között fennforgó legnehezebb és legártalmasabb két adversa religiák ellenében plántáltatott volt a boldog emlékezetű fejedelmektől, az a jó véget a patrociniumnak elégtelensége miatt mindeddig is el nem érte, hanem sőt inkább csufolásra adott sokakban azoknak kész alkalmatosságot, az alkalmatlan tanittások és tanulások által."

A másik idézet szintén egy levél, amelyet 1817. január első napjaiban Bolyai Farkas írt Bodor Pálhoz:

„Kihozattam volt az oda beküldött piece-ket: sokat corrigáltam bennek, mert a historica fides ellen iszonyuan hibáztam volt és nem olvastam a criticát ..."

A két levél között több mint 150 év telt el, ennek ellenére a két írásmód között nincs nagy különbség. Felgyorsult életünkben talán ennél kevesebb idő is elég lesz ahhoz, hogy az informatika latinjából — az angolból — a nyelvünkbe „ragadt szószörnyszülötteket" magyarul is jól hangzó és a fogalmat is hűen visszaadó szavakkal „corrigáljuk". Ha nem tesszük, akkor „iszonyúan hibázunk" nyelvünk ellen, és jogosan kapjuk a megérdemelt „criticát".

KOVÁCS GYŐZŐ

PÁLYÁZAT

közoktatási számítógépes programok elkészítésére

Közöljük a Tudományszervezési és Informatikai Intézet módosított oktatási programpályázatát, amely felváltja a korábbit. Felhívjuk olvasóink figyelmét a lényeges eltérésekre. Az új pályázat közoktatási, a korábbi középiskolai helyett, tehát általános iskolákra is kiterjesztették. A gépek köre, amelyre programot lehet írni, szélesedett: Commodore 64, C16 és Primo is alkalmazható. A programok megkövetelt színvonalát magasabbra helyezik, és követelmény, hogy azok az eddig ismertekhez képest újszerűek legyenek.

A Művelődési Minisztérium megbízásából a Tudományszervezési és Informatikai Intézet (a továbbiakban TII) pályázatot hirdet közoktatási intézményekben használható számítógépes oktatási programok elkészítésére.

A pályázat célja olyan oktatási programcsomagkészlet létrehozása, ami lehetővé teszi, hogy a közoktatási intézmények oktatási feladataik teljesítésébe bevonják a számítógépet.

A pályázat tárgya olyan HT—1080Z, HT—1080Z/64, Commodore 64, Primo, Commodore 16, illetve egyéb típusú, e pályázatot követően központilag biztosított mikroszámítógépre készített *magas színvonalú*, újszerű oktatási program, amelyik szervesen illeszkedik az alapfokú, illetve a középfokú oktatási intézmények tananyagához, munkájához, és/vagy lehetővé teszi a mikroszámítógép alkalmazásának bemutatását, *és amelyhez hasonló a forgalmazott oktatási programok között nem található*. Az alapfokú programokat jelenleg Commodore 16-ra és Primóra, középfokút HT-kre, Commodore 64-re és Primóra lehet készíteni.

Pályázni lehet az alábbi típusú programokkal:

1. az oktatást segítő (valamely tantárgyhoz kapcsolódó) program,
2. a számítástechnika oktatását segítő program,
3. bemutató program (demonstrációs program),
4. oktatáshoz kapcsolódó, didaktikus játékprogram,
5. egyéb, a mikroszámítógép alkalmazását (pl. iskolai adminisztrációt stb.) elősegítő program.

Pályázni csak a pályázó, illetve pályázók szellemi tulajdonát képező — új, forgalomba még nem hozott — önálló oktatási programmal lehet. A pályázatnak tartalmaznia kell: a pályázó(k) nevét, munkahelyét, lakcímét, a beküldött programok nevét, a mellékletek listáját.

A mellékletek a következők:

a) nyilatkozat, csoportos pályázat esetén a tulajdonjog elosztásával együtt,

b) programismertető(k),

c) programleírás(ok),

d) programo(ka)t tartalmazó mágneskazetta(k).

A b), c), d) részeket az IHE 002 jelű „Középiskolai számítógépes oktatási program készítésének előírásai" c., az előző pályázathoz kiadott anyag alapján kell elkészíteni. A „Programleírás" követelményei differenciáltak, egyes részeit elegendő a pályázat elfogadása után, de megvétele előtt beküldeni.

Egy pályázat több oktatási programra is vonatkozhat, ekkor minden oktatási programhoz mellékelni kell a fentiekben felsorolt dokumentumokat. Egy kazettára több program is rögzíthető, ez esetben a mágneskazetta borítóján közölni kell a rögzítés sorrendjét.

Pályázatot bármikor be lehet küldeni az alábbi címre: Tudományszervezési és Informatikai Intézet, Budapest, Pf. 454. 1372.

Pályázatot bárki benyújthat, aki kedvet érez közoktatási intézményekben használható program készítésére. Pályázni egyedileg vagy csoportosan is lehet. Csoportos pályázat esetében minden pályázónak alá kell írni a nyilatkozatot, feltüntetve a szerzői jog megoszlását.

A beérkezett pályázatokat a TII ellenőrzi komplettség és működőképesség szempontjából. Hiányosan beküldött pályázatok, illetve nem működő programok nem kerülnek elbírálásra, ezeket a TII visszaküldi a pályázóknak. A pályázatok beérkezését a TII 8 napon belül nyugtázza.

A pályázatokat a Művelődési Minisztérium által felkért bíráló bizottság bírálja el használhatóság, hasznosság és érték szempontjából. A bíráló bizottság negyedévenként egyszer ülésezik.

Az elbírálásnál előnyben részesülnek azok a pályázatok, amelyek konkrét tananyaghoz kapcsolódnak. A TII a pályázót a bizottság ülését követő 15 napon belül értesíti a döntésről. *Az elutasító döntés ellen fellebbezésnek helye nincs.*

Pályadíj nincs. Az elfogadott és a bíráló bizottság esetleges észrevételei alapján kiegészített pályázatokra a Tudományszervezési és Informatikai Intézet vételi ajánlatot tesz. Elfogadása esetén a program az intézet kizárólagos tulajdonába megy át, amely jogosult ezt a pályázónak fizetett minden további ellenszolgáltatás nélkül közvetlenül vagy megbízott útján belföldön forgalmazni. Külföldi értékesítésnél a forgalmazó új szerződést köt a pályázóval. Az elutasított pályázatot a TII visszaküldi a pályázaton 1. helyen szereplő pályázónak.

A pályázat folyamatos, visszavonásig érvényes.

Az említett, IHE 002 jelű anyag a TII-től igényelhető postai úton vagy személyesen az alábbi címen:

Tudományszervezési és Informatikai Intézet, Budapest XI., Egry József u. 1—9. „E" ép. XI. em. 5. szoba.

E pályázat kihirdetésével egyidejűleg az 1983. május 20-án kiírt és 1983 decemberében módosított „Középiskolai számítógépes oktatási program" pályázatot visszavonom.

PÁRIS GYÖRGY sk.
a Tudományszervezési és Informatikai Intézet igazgatója

Gép a gépen II.

Programunk második részében tovább építgetjük a szubrutinokat, amelyekből a már látott ábrák összerakhatók.

Eddig vonalakat, dobozokat rajzoltunk, és elkezdtük a BUS-vonalak kiépítését. Most a BUS-vonalak függőleges részének felrajzolásával kezdjük, a 400-as sorszámtól TELÍTÉS TÖRLÉS FEL kezdő megjegyzéssel. Ehhez a PRÓBA5 tesztelőprogram tartozik. Az 5000-as sorszámon bevitt adatok a telítés, az 5080-as sor adatai a törlést végzik. A szubrutin megadott hosszúságú és szélességű függőleges szalagot rajzol, majd töröl.

```
390 REM**********************************
**********
400 REM TELITES TORLES FEL  SD OD HD SZ S
T T
410 HV=SZ:OV=OD:IFST=-1THEN440
420 IFNOT(T)THENK1=1:K4=133:K2=4:K5=133:
K3=6:K6=149ELSEK1=5:K4=148:K2=3:K5=144:K
3=7:K6=32
430 GOTO450
440 IFNOT(T)THENK1=3:K2=5:K3=6:K4=144:K5
=148:K6=149ELSEK1=4:K2=1:K3=7:K4=133:K5=
129:K6=32
450 FORJ=SDTOSD+HD*STSTEPST
460 SV=J:KV=K1:GOSUB60:PRINTEP+SZ,CHR$(K
4);:KV=K2:GOSUB60:PRINTEP+SZ,CHR$(K5);:K
V=K3:GOSUB60:PRINTEP+SZ,CHR$(K6);
470 NEXT:RETURN
480 REM**********************************
**********
490 REM NYIL ES TORLES
500 REM NYIL BALRA JOBBRA SN O1 JB D T
510 IF(T)THEN560
520 FORI=DTO0STEP-1
530 SET(O1-2*JB*I,SN+I):SET(O1-2*JB*I,SN
-I):SET(O1-2*JB*I+JB,SN+I):SET(O1-2*JB*I
+JB,SN-I)
540 NEXT:RETURN
550 REM NYIL TORLES BALRA JOBBRA SN O1 J
B D T
560 FORI=DTO0STEP-1
570 RESET(O1-2*JB*I,SN+I):RESET(O1-2*JB*
I,SN-I):RESET(O1-2*JB*I+JB,SN+I):RESET(O
1-2*JB*I+JB,SN-I)
580 NEXT:RETURN
590 REM**********************************
**********
600 REM NYIL FEL SN O1 D JB T
610 IFTTHEN650
620 FORI=DTO0STEP-1
630 SET(O1-I,SN+I*JB):SET(O1+I,SN+I*JB)
640 NEXT:RETURN
650 REM NYIL TORLES FEL SN O1 D
660 FORI=DTO0STEP-1
670 RESET(O1-I,SN+I*JB):RESET(O1+I,SN+I*
JB)
680 NEXT:RETURN
690 REM**********************************
**********
700 REM BEIRAS FUGGOLEGESEN A$ SI OI
710 FORI=1TOLEN(A$):PRINTE(SI-1+I)*64+OI
,MID$(A$,I,1);:NEXT:RETURN
720 REM**********************************
**********
730 REM VEZERLO VONAL ES TORLESE SD OD S
Z T
740 OD=11:SZ=10
750 FORI=ODTOOD-SZSTEP-1:SV=SD:OV=I:HV=O
D-I:KV=K1:GOSUB60:NEXT:PRINTESD*64+OD-SZ
/2,B$;:PRINTE(SD-1)*64+OD-SZ/2,A$;:RETUR
N
760 IFNOT(T)THENB$="(":K1=4:GOSUB740ELSE
B$=" ":A$="     ":K1=7:GOSUB740
770 RETURN
780 REM**********************************
**********
790 REM HANGHOZ
800 FORH=0TO15:OUT31,H:OUT30,A(H):NEXT:R
ETURN
810 REM**********************************
**********
820 REM HANG MAX
830 A(8)=15:A(9)=15:A(10)=15:GOSUB800 :R
ETURN
840 REM**********************************
**********
850 REM HANG MIN
860 A(8)=0:A(9)=0:A(10)=0:GOSUB800:RETUR
N
```

```
5000 REM PROBA5
5010 PRINT" TELITES TORLES FEL ..."
5020 PRINT"HOVA TEGYEM":INPUT"FUGGOLEGES";SD
5030 INPUT "VIZSZINTES";OD
5040 INPUT"HOSSZA";HD
5050 INPUT"SZELESSEGE";SZ
5060 ST=1:G=1:T=0
5070 CLS:GOSUB400
5080 ST=1:G=1:T=-1:GOSUB400
5090 REM EGY SATIROZOTT TEGLALAPOT KAPUNK,
     AMELY FENTROL LEFELE TELITODIK MAJD FOGY.
5100 REM SD=5, OD=10, HD=10, SZ=10 ADATOKKAL
     LEHET KIPROBALNI
5110 REM ST=-1-re AZ IRANY MEGFORDUL.
     EKKOR SD=10 LEGYEN

5000 REM PROBA6
5010 PRINT" NYIL BALRA JOBBRA ...."
5020 PRINT"HOVA TEGYEM":INPUT"FUGGOLEGES";SN
5030 INPUT "VIZSZINTES";O1
5060 JB=1:D=5:T=0
5070 CLS:GOSUB500
5075 IFINKEY$=""THEN5075
5080 JB=1:D=5:T=-1:GOSUB500
5090 REM EGY JOBBRA MUTATO NYILAT KAPUNK,
     MEGNYOMASA UTAN ELTUNIK.
5100 REM SN=10, O1=10 ADATOKKAL LEHET KIPROBALNI
5110 REM SN=-1-re AZ IRANY MEGFORDUL.
```

A BUS-vonalakon haladó információ irányát nyíl mutatja. Az 500-as sor kezdi ezt megrajzolni, az 550-es pedig törli. A PRÓBA6 mindkettőt teszteli. A 600-tól kezdődő szubrutint és a PRÓBA7 programot nem szükséges külön részletezni.

Ábráinkba több helyen kell függőlegesen beírni. Ezt a 700-as sor indítja el. Bemenő adatai SI, OI kezdő koordináták és a beírandó A$. Kipróbálását az olvasóra bízzuk.

A vezérlő vonalak rajzolása a 740-es soron kezdődik. Tesztelő program helyett csak annyit írunk, hogy a 760-as számú szubrutint kell meghívni, miután a T=0 vagy T=−1 értéket állítottuk be, illetve A$-nek 4 karakter hosszúságú értéket adtunk.

A hang beállítását nem részletezzük. Annyi látható, hogy az A (15) vektor elemeinek egy részét fixre beállítjuk, más részüket változtatjuk.

NYIRATI LÁSZLÓ

Számítástechnikai tanfolyam egy amerikai iskolában

Egy amerikai iskolában szabadon választható számítástechnikai tanfolyamot rendeztek. Az erről szóló beszámolót Marika Torok 10 éves, 4. osztályt végzett tanuló elmondása és az általa elkészített anyagok alapján állítottam össze.

A jelentkezők közül tesztvizsgán választották ki az alkalmasakat. A vizsga az általános ismereti színvonal meghatározására irányult. Feladat volt például az állam fővárosának kiválasztása 3 városnévből, egy szorzás eredményének kiválasztása 3 megadott szám közül stb. A két 4. osztály 45 tanulójából 6 felelt meg a követelményeknek.

A 4. osztályosoknak heti egy alkalommal, hat héten át, alkalmanként kb. egy óra hosszat tartott a tanfolyam. A vizsgára 3-6. osztályosok jelentkezhettek. A felsősök tanfolyama hosszabb ideig tartott, de annak anyagát Marika nem ismerte.

Egy-egy tanár 6-10 gyerekkel foglalkozott. Minden gyereknek külön PET gépe volt, magnetofonnal, monitorral. Minden gyerekpár kapott egy nyomtatót. A tanárnak másfajta gépe volt, két hajlékonylemez-egységgel.

A tanfolyam *első napján* a tanár beszélt a számítógépről, a programról. Gépet nem mutatott. A gyerekek házi feladata képek, cikkek gyűjtése volt.

A *második napon* a tanár elmondta, hogy mit adjanak be a

1. ábra

A1

Name Marika Torok

Arithmetic Operations + - * /

① NEW
```
10 print " addition problem"
20 print "9+6"
30 print 9+6
40 print "9+6=" 9+6
50 end

LIST
RUN
```

② NEW
```
10 print "subtraction problem
20 print "9-6"
30 print 9-6
40 print "9-6=" 9-6
50 end

LIST
RUN
```

③ NEW
```
10 print "multiplication problem"
20 print "8*4"
30 print 8*4
40 print "8*4=" 8*4
50 end

LIST
RUN
```

④ NEW
```
10 print "division problem"
20 print "8/4"
30 print 8/4
40 print "8/4=" 8/4
50 end

LIST
RUN
```

2. ábra

A2

Arithmetic Operations

① 10 PRINT 25+5 → 30
20 PRINT 25-5 → 20
30 PRINT 25*5 → 125
40 PRINT 25/5 → 5
50 END

② 10 PRINT " ♥ "
20 PRINT " 75 + 10 = "
30 PRINT 75+10
40 PRINT " 65 - 15 = "
50 PRINT 65-15
60 END

Ready
75+10=
85
65-15=
50

③ CHECKING ANSWERS

26*5=
130

130/5=
26

```
10 Print "Checking Answers"
20 Print
30 Print "26*5="
40 Print 26*5
50 Print
60 Print "130/5="
70 Print 130/5
80 End
```

3. ábra

A3

Arithmetic Operations (cont'd)

④ Equaling 100

10 * 10 =
100

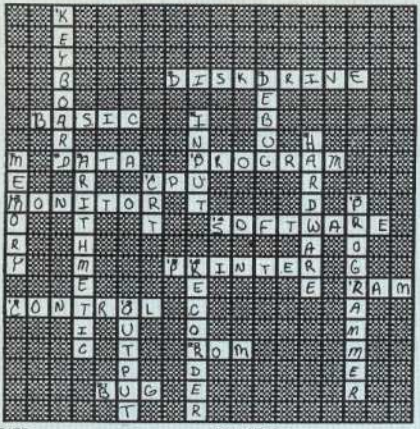
25+75 =
100

125 -25 =
100

200 / 2 =
100

(clear screen)
```
10 Print " ♥ "
20 Print "Equaling 100"
30 Print
40 Print " 10*10="
50 Print 10*10
60 Print
70 Print "25+75="
80 Print 25+75
90 Print
100 Print " 125-25="
110 Print 125-25
120 Print
130 Print "200/2="
140 Print 200/2
150 End
```

⑤ Writ a program which shows the nine times table. Clear screen first. Show the calculation on one line first and the answer on another. Skip a line between facts.



CROSS CLUES

2. AN INPUT/OUTPUT DEVICE THAT USES FLOPPY DISKS
4. A COMPUTER LANGUAGE
8. INFORMATION
10. A SET OF INSTRUCTIONS THAT TELLS THE COMPUTER WHAT TO DO
11. CENTRAL PROCESSING UNIT
12. SCREEN
14. PROGRAMS
15. AN OUTPUT DEVICE THAT PRINTS OUTPUT ON PAPER
17. THE COMPUTERS TEMPORARY MEMORY WHERE PROGRAMS AND DATA ARE STORED
18. THE _____ UNIT OF THE CPU DIRECTS THE COMPUTER SYSTEM
20. READ ONLY MEMORY—THE COMPUTER'S PERMANENT MEMORY
21. A MISTAKE IN A PROGRAM

DOWN CLUES

1. AN INPUT DEVICE USED FOR TYPING
3. TO CORRECT A PROGRAM
5. INFORMATION TO BE PUT INTO THE COMPUTER
6. COMPUTER MACHINERY
7. THE PART OF THE CPU THAT STORES DATA
9. THE _____—LOGIC UNIT IN THE CPU DOES CALCULATIONS AND COMPARES
11. SCREEN
13. A PERSON WHO WRITES COMPUTER PROGRAMS
16. AN CASSETTE _____ IN AN INPUT/OUTPUT DEVICE THAT STORES DATA AND PROGRAMS ON CASSETTE TAPE
19. THE RESULTS OF A PROCESSED PROGRAM

4. ábra

S2

Name Marika Torok

```
10 PRINT "■"
20 PRINT "I ";
30 PRINT "AM ";
40 PRINT "A ";
50 PRINT "PET ";
60 PRINT "COMPUTER."
```

I am a PET COMPUTER.

```
10 PRINT "Calculating"
20 PRINT
30 PRINT "15*3 = ";
40 PRINT 15*3
50 PRINT "45/3 = ";
60 PRINT 45/3
70 END
```

Calculating

15*3 = 45

45/3 = 15

③

1	2	3
*	**	***
first	Second	Third

```
10 PRINT " 1; 2, 3"
20 PRINT " *, **, ***"
30 PRINT "First", "Second", "Third"
40 END
```

④ Write a program which displays the numbers from one to ten in a column and their names in the next column.

6. ábra

5. ábra

S1

Name

Spacing

	Output
PRINT 1	1
PRINT 1,2	1 2
PRINT 1,2,3	1 2 3
PRINT 1,2,3,4	1 2 3 4
PRINT 1,2,3,4,5	1 2 3 4 5
PRINT 1,2,3,4,5,6	1 2 3 4 5 6
PRINT 1;2;3	1 2 3
PRINT 1;2;3;4;5;6;7	1 2 3 4 5 6 7
PRINT "A","B","C","D"	A B C D
PRINT "A";"B";"C";"D";"E";"F"	ABCDEF

```
10 PRINT 1,2,3,
20 PRINT 4
30 END
```

1 2 3 4

```
10 PRINT 1;2;3;
20 PRINT 4;5
30 END
```

1 2 3 4 5

```
10 PRINT "A","B";
20 PRINT "C";"D"
30 PRINT "E";"F";"G";"H"
40 END
```

A B C D E F G H

7. ábra

billentyűzeten keresztül. Ezek egyszerű PRINT utasítások voltak, mint például a feladatlap 1. feladata *(1. ábra)*, amely az alapműveletek végzését, szövegek kiíratását gyakoroltatta. Azok a gyerekek, akik már használtak gépet, a billentyűket gyorsan megtalálták, de a többiek nem lettek aznap készen.

Harmadik nap: az előző napi anyag folytatása. A tanár a gyerekek kérésére segített.

Negyedik nap: játék. A gyerekek INPUT, OUTPUT, CPU, MEMORY, Arithmetic Unit elnevezésű szereplőkkel gyakorolták be a számítógép működését. Például egy papírlapra felírtak egy feladatot, azt odaadták az INPUT-nak, az továbbította a CPU-nak stb. Ezután a tanár programokat mutatott be, többek között zenélt.

Az *ötödik napon* a feladatlap A2-A3. oldalán *(2., 3. ábra)* levő feladatokat oldották meg. Könyvekből ismétlő feladatokat kaptak, és megoldották a keresztrejtvényt *(4. ábra)*. Otthon összeállítottak egy újságkivágásokat tartalmazó könyvet.

MARIKA TOROK

READY.

```
 5 OPEN 1,4 CMD 1
 10 PRINT " MARIKA TOROK      "
 20 PRINT
 30 PRINT
 40 PRINT
 50 PRINT "
 60 PRINT "■
 70 PRINT "■
 80 PRINT "■
 90 PRINT "■
 100 PRINT"■
 110 PRINT"■
 120 PRINT"■
 130 PRINT"■
 140 PRINT"■
 150 PRINT"■
 160 PRINT"■
 170 PRINT"■
 180 PRINT"■
READY.
```

A *hatodik napon* a feladatlap vizsga része került sorra *(S1-S2, lásd 5., 6. ábra)*. Megoldották a könyvecskéhez kapcsolt vizsgafeladatokat. Egy külön nap szolgált a további feladatok megoldására. Ezek közül bemutatunk kettőt: a

TRS-80 June 23, 1985

things to do today

date _____ 19____

```
5 Print "Good - bye!"
10 sound 40,10
20 sound 4,10
Date - 24-85
5 cls( 7)
10 set (32,13,8)
20 for H,1 28T036
30 set (H,16,5 )
40 Next H
50 set 05,10 ,31
60 set 08,19,32)
70 Print "HELLO!!
80 sound 39,7
90 sound 3,5
100 End
Date 2485
80 sound 4,10
90 sound 25,5
100 Sound 13,7
110 Sound 78,9
120 sound 20,7
130 sound 4,10
140 Sound 12,6
150 sound 22
160 sound 90,9
170 sound
180 sound 1,5
190 Goto 190
200 end
24-85
5 sound 4,10
10 sound 13,10
20 sound 23,9
30 sound 45,10
40 sound 34,10
50 sound 4,9
addade 24-85
60 sound 4,10
70 sound 13,10
80 sound 23,8
90 sound 45,10
100 sound 34,10
105 sound 7,9
```

8. ábra

programtervezést és a nyomtatást *(7. ábra)*.

A kislány szüleitől, akik jónevű számítástechnikai szakemberek, egy TRS 80 MODEL III típusú gépet kapott gyakorlásra. A tanfolyammal egy időben, a harmadik óra idején elkészített első zenei programját is bemutatjuk *(8. ábra)*. Látható, hogy a tanultakon kívül már a CLS, SET, FOR... NEXT, GOTO, SOUND utasításokat is használta.

Megjegyezzük, hogy az itt is említett, *számítógép nélkül számítástechnikát oktató módszert*, amivel például a LOGO jól tanítható, nálunk is célszerű lenne elterjeszteni.

DR. SIMONYI ENDRE

Egy tisztán szoftver úton megvalósított beszédszintetizátor

A Burgaszi (Bulgária) Matematikai Gimnázium három 10. osztályos tanulója a BUSOFT Szoftverház részére kidolgozott egy 30 kbájt összterjedelmű beszédszintetizáló programcsomagot. Az ő elmondásuk alapján számolok be munkájukról.

A cél nem kiváló minőségű, hanem jól reprodukálható hang létrehozása volt. A reprodukciós hiba közelebbről meg nem határozott körülmények között 20-25%, ha az emberi beszédet vesszük alapnak. Semmiféle hardvermódosítást, kiegészítést nem alkalmaztak.

Az analóg jelet digitalizálják, majd tárolják. Ez utóbbihoz 6-8 kbájtot használnak. A megoldás jóságához elengedhetetlenül szükséges a nagy feldolgozási sebesség, mivel ez szabja meg a maximális átviteli frekvenciát. A gyűjtött adatokat továbbfeldolgozás előtt lemezre viszik. A hangképzés 1 bites digitál/analóg átalakítóval történik. Ez határozza meg a hangminőséget.

A lemezre vitt digitális jelek továbbfeldolgozását, szerkesztését nem feltétlenül kell ugyanazzal a rendszerrel végezni. Elmondták, hogy az eddigi legjobb minőségű angol szöveghangot egy meg nem nevezett típusú, Hewlett-Packard gyármányú számítógéppel állították elő, bár a program Apple II+ kompatibilis gépekre íródott.

A feldolgozást bájtonkénti szerkesztéssel-javítással végezhetik, ami lényegesen jobbá teszi a minőséget. Így például lehetővé tesz szoftveres zajszűrést, kiküszöbölhetővé teszi a szavak kiejtésének különböző sebessége miatt fellépő frekvenciakülönbséget, lehetővé teszi a mondatok, szavak, hangok elkülönítését.

A módszer a hangok különkülön történő előállítása miatt korlátlan szókészletű, és nem mereven nyelvhez kötött. Leírtam például cirill betűkkel ezt a szöveget: „A Valcsev-féle program bemutatkozik a magyar rádióhallgatóknak. A hanghatás ugyan nem túl jó minőségű, de ennek oka, hogy nem használ semmiféle hardver kiegészítőt." A kimondott szöveg határozottan érthető volt.

A szerkesztési lehetőségeken kívül a frekvencia-amplitúdó görbék is kijeleztethetők. Ezek figyelembevételével a reprodukciós képesség a szerkesztés révén elérhető 8-10%-ról 28-35%-ra növelhető. A program készítői úgy vélik, hogy ez a módszer a tisztán szoftveres módszerek nyújtotta lehetőségeket legjobban kihasználja.

Mivel a szoftveres lehetőségek korlátozottak, igazán jó hangminőséget az alacsony (mintegy 7 kHz) vágási frekvencia miatt biztosítani nem tudnak, ezért hardveres kiegészítők használatát tervezik.

A program étlaprendszerű. A fő- és mellékétlap az *1.* és *3. ábrán*, a grafikus kijelzés kiválasztása a *2. ábrán* látható. A nyomtató latin betűs volt, így egyes cirill betűk helyett írásjeleket, például zárójelet írt.

Dr. S. E.

```
]
]RUN
▙▙
▙2
GLAWNO MEN@
´OSW´- MEN@ //
´R´- GRAFIKA NA DANNITE
´S´- ZAPIS NA TEKU]ITE DANNI
´L´- ZAREVDANE NA DANNI OT DISK
´D´- OBRABOTKA NA NOWI DANNI
´I´- OBSLUVWA]I PROGRAMI
´H´- KRAJ
IZBERETE VELANATA OT WAS OPCIQ...
```

1. ábra

```
ZAREVDANE...


▙▙▙2
▙
GRAFIKA NA DANNITE
==================================
´MK+C´-WRY]ANE W GLAWNO MEN@
DESET.NA¬.ADRES-8192
▙▙
```

2. ábra

3. ábra

```
]
]RUN
▙▙▙2
▙▙2▙
MEN@ //
´E´-GRAFI¬EN REDAKTOR
´S´-NABL@DAWANE NA ¬AST OT GRAFIKATA
´M´-POMO]NA TABLICA
´P´-PROSLU[WANE NA DANNITE
´OSW´-GLAWNO MEN@
´V´-MA]ABEN REDAKTOR
´C´-SPECIALNI FUNKCII
IZBERETE VELANATA OPCIQ...
```

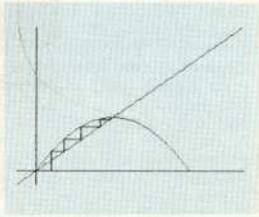
Káosz a Spectrumban

1985/4. számunkban közöltük Major Zoltán Egyenletek közelítő megoldása című cikkét. A közelítő vagy más néven iterációs eljárások röviden a következők. Keressük az $f(x)=0$ egyenlet megoldását. A zérushely keresését visszavezetjük az $f(x)+x=g(x)$ függvény és az $y=x$ egyenes metszéspontjának keresésére, ezt hívják fixpontnak. (Bizonyos esetekben azonban az $x+c\cdot f(x)=g(x)$ függvény hatásos.) A vizsgált intervallum egy tetszőleges x_0 értékével kezdődik a közelítés, majd $x_1=g(x_0)$, $x_2=g(x_1)$ ... $x_n=g(x_{n-1})$ sorozattal keressük a fixpontot. Ha ez konvergens, vagyis tart x_p-hez, akkor $x_p=g(x_p)$ és $f(x_p)=0$, tehát megvan a gyökhely. A konvergencia kritériuma, hogy a fixpont környékén ne legyen „túl meredek" a g(x) függvény.

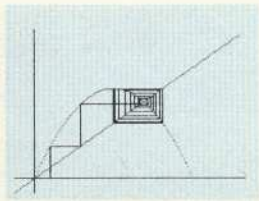
Jelen cikkünk a divergens esetekkel foglalkozik. A divergenciavizsgálatban találkozhatunk napjaink tudományának egyik olyan kutatási területével, amelyet éppen a számítógépes kísérletek indítottak el, s a számítógépes módszerekkel még további eredményeket hozhatnak. E területnek, az ún. nem lineáris rendszerek dinamikájának első pillantásra is lenyűgöző eredményei könnyen és látványosan vizsgálhatók személyi számítógépek segítségével.

Gyakran egy folyamatot rendszerparaméterektől is függő képlettel jellemezhetünk. Például egy ágyúval a puskapor mennyiségét és a lövés szögét megfelelően változtatgatva elérhetjük, hogy a kilőtt golyók közelítőleg az $f(r,x)=r\cdot x\cdot(1-x)$ parabola szerinti pályákat fussanak be. Az r-et kontrollparaméternek szokták nevezni, érkétől függően laposabb vagy meredekebb a görbe. Bizonyos differenciálegyenletek megoldása során is előfordul a fenti összefüggés.

Ha $0<r\leq 4$, akkor az $f(r,x)=r\cdot x\cdot(1-x)$ függvény a [0,1] intervallumot önmagába képezi le, folytonos, tehát létezik fixpontja. Az



1. ábra (r= 2)

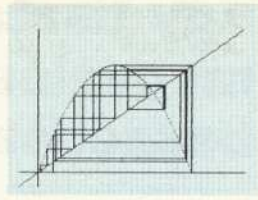


2. ábra (r= 1+√5)

$x=r\cdot x\cdot(1-x)$ egyenletből kapjuk az $x_1^*=0$ és az $x_2^*=(r-1)/r$ megoldásokat. Ha az $x_{n+1}=r\cdot x_n\cdot(1-x_n)$ iterációval akarjuk ezeket meghatározni, akkor tapasztalhatjuk (és elméletileg is bizonyítható), hogy $0<r\leq 1$ esetén az iteráció az $x_1^*=0$-hoz, míg $1<r\leq 3$ esetén az $x_2^*=(r-1)/r$-hez konvergál *(1. ábra)*.

Ha $3<r\leq 4$, akkor az x_2^* pontban a függvény érintőjének meredeksége $m_{x_2^*}=2-r$, azaz $|m_{x_2^*}|>1$, tehát az iteráció nem lehet konvergens. Viszont a [0,1] intervallumból kilépni sem tudunk, így a divergencia csak valamiféle oszcilláció lehet! Ennek tulajdonságait és jelentőségét az utóbbi húsz évben tárták fel. Fantasztikusan új és meglepő eredmények ezek, figyelembe véve az iterációs eljárások közismertségét, és azt a véleményt, mely szerint ezen a területen már minden lényeges eredmény ismert.

Tekintsük az $x_{n+1}=r\cdot x_n\cdot(1-x_n)$ iterációt, amikor r egyre jobban meghaladja a 3-at. Ha $r_1\equiv 3<r\leq(1+6)\equiv r_2$, akkor néhány iteráció után két érték fog váltakozni, tehát kialakul egy 2 periódusú határciklus. A *2. ábrán* ez látható,



3. ábra (r= 4)

4. ábra

```
 10 PROC Main
 20 DEF PROC Main
 30   PROC Koord
      PROC YX
      PROC Graf
      PROC Iter
 40 END PROC
 50 DEF PROC Koord
 60   RESTORE 400
 70   READ xos,yos,xrg,yrg
 80   PLOT -xos,0
      DRAW xrg-1,0
 90   PLOT 0,-yos
      DRAW 0,yrg-1
100 END PROC
110 DEF PROC YX
120   RESTORE 410
130   READ dx,dy
140   LET mn=xos/dx
      LET w=yos/dy
150   IF w<mn THEN LET mn=w
160   LET mx=(xrg-xos-1)/dx
170   LET w=(yrg-yos-1)/dy
180   IF w<mx THEN LET mx=w
190   PLOT -dx*mn,-dy*mn
      DRAW dx*(mn+mx),dy*(mn+mx)
200 END PROC
210 DEF PROC Graf
220   RESTORE 420
230   READ f$
      READ xa,xb
240   FOR i=dx*xa TO dx*xb
250     LET x=i/dx
        LET j=dy*VAL f$
260     PLOT i,j
270   NEXT i
280 END PROC
290 DEF PROC Iter
300   RESTORE 440
310   READ n
      READ x
320   LET i=dx*x
      LET j=0
330   PLOT i,j
340   FOR l=1 TO n
350     LET x=VAL f$
        LET ij=dy*x
360     DRAW 0,ij-j
        LET ji=dx*x
370     DRAW ji-i,0
        LET j=ij
        LET i=ji
380   NEXT l
390 END PROC
400 DATA 15,15,256,176
410 DATA 120,120
420 DATA "4*x*(1-x)"
430 DATA 0.1
440 DATA 20
450 DATA 0.1
```

$r=1+\sqrt{5}$. Ha $r_2\equiv 1+\sqrt{6}<r\leq r_3$, akkor az iteráció során már négy érték ismétlődik, azaz egy 4 periódusú határciklus jön létre. Tovább növelve r-et, rendre 2^k értékből álló határciklushoz konvergál az iteráció egy (r_k, r_{k+1}) intervallumon. Ennek a hossza zérushoz tart, s kimutatható, hogy a $\lim_{k\to\infty} r_k=r_\infty$ határérték is létezik. (r_∞ számítógépes meghatározása 3,569945672 közelítő értéket adott.) Ha $r=r_\infty$, akkor megszámlálhatóan végtelen sok ponton fut végig az iteráció!

Ha az r-et r_∞ fölé növeljük, a helyzet egyre bonyolultabb lesz. Bizonyos r-ekre az iteráció véges sok, de nem feltétlenül 2^k pontból álló határciklushoz tart, más r-ekre viszont (és ezekből nagyon sok van) nem periodikus, kaotikus sorozatot alkot. Ezen azt értjük, hogy míg a konvergens iterációnál n nagyobb értékeire biztosan tudjuk, hogy x_n a fixpont egyre kisebb környezetében lesz, addig a kaotikus sorozatoknál csak azt tudjuk, hogy $x_n\in[0,1]$, **és hogy egyenlő valószínűséggel esik ugyanolyan hosszú részintervallumokba.** Például az $r=4$ esetben is kaotikus sorozatot hoz létre az iteráció. A *3. ábrán* ezt mutatjuk be.

A fentiekben vázolt, a rendszerparaméterek bizonyos értékeinél létrejövő kaotikus sorozatok olyan univerzális törvényszerűségeket mutatnak, amelyek függetlenek attól, hogy milyen fizikai, kémiai vagy egyéb rendszer vizsgálatakor bukkantunk rájuk. Ennek oka valószínűleg a nemlineáritások természetében rejlik. Ezek vizsgálata, törvényszerűségeinek feltárása intenzív számítógépes módszereket követel.

Az iterációkat rajzoló programot *(4. ábra)* Spectrum gépre Betabasic nyelven írtam. Nem célom most ennek az ismertetése, úgy gondolom, a közölt lista alapján érzékelhető, hogy egy jól használható, valamelyest már strukturált kód létrehozására alkalmas szoftver.

A bemenő adatokat az egyszerűség kedvéért DATA sorokban helyeztem el, természetesen ki-ki ízlésének megfelelő interaktív változatot is készíthet belőle. Felhívom a figyelmet arra, hogy az iterációban szereplő függvény a Spectrum jóvoltából sztringváltozó.

LOVRICS LÁSZLÓ

Alapozás XVI.

Rendszerek „élete", működése állapotátalakulási folyamataikkal írható le. Megismerkedtünk néhány hasznos ábrázolási eljárással, amelyek a felbecsülhetetlen értékű állapottérrel kapcsolatosak. Miután a szemléltető eszközök a kezünkben vannak már, az egymás után következő rendszerállapotok ábrázolásával, azaz például azzal foglalkozunk, hogy hogyan mozognak az egyes rendszereket képviselő pontok az állapottérben.

Kívánalmak és kielégítési módjaik

Nemcsak az ember viselkedését befolyásolják saját igazi vagy másoktól átvett, sőt kívülről rákényszerített kívánságok. Ha figyelmesek vagyunk, minden jelenség mögött észrevehetjük azokat a kívánalmakat, amelyeknek a megfigyelt jelenség kielégítése.

A számológépben (és minden digitális automatában) ezek a kívánalmak többfélére vonatkoznak (például információtárolásra, információtovábbításra, információkból más információk származtatására, információmegjelenítésre).

Bármivel foglalkozunk is, a megértés kulcsa, hogy biztos áttekintésünk legyen a kívánalmakról, azok kielégítési eszközeiről és módjairól, és az összes szereplő lényeges tulajdonságairól.

A *kívánalom* a matematikai és számítástechnikai munkában is a legfontosabb fogalmak közé tartozik. Aki csinálta is ezeket a tevékenységeket, nemcsak előadást tartott vagy csak könyvet írt róluk, annak tapasztalata igazolja állításunkat.

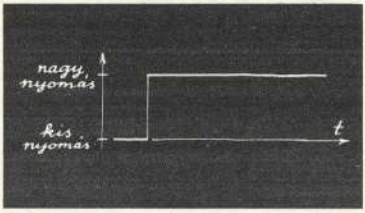
Kívánalmak csak papíron és íróasztal mellett fordulnak elő egyesével és egymástól elszigetelten. A gyakorlati munkában mindig *kívánalomrendszerekkel* állunk szemben. Ezzel kapcsolatban három eset lehetséges. A legritkább az, amikor a kívánalomrendszer *kielégíthető, éspedig egyféleképpen.* Ennél sokkal gyakoribb a másik két eset, az, amikor a kívánalomrendszer *nem elégíthető ki* úgy, hogy minden kívánalomnak (kívánságnak) teljes mértékben eleget tegyünk, és az, amikor *minden kívánalom kielégíthető, és ennek több módja is van.*

Érdemes elgondolkodni azon, hogy miért mellőzik a tankönyvek a gyakorlat számára nélkülözhetetlen két utóbbi eset tárgyalását. Ezek részletes elemzésébe — sajnos — mi sem bocsátkozhatunk, de mivel fejtegetésünk gyakorlati igyekszik lenni, nem kerülhetjük meg a témát szó nélkül.

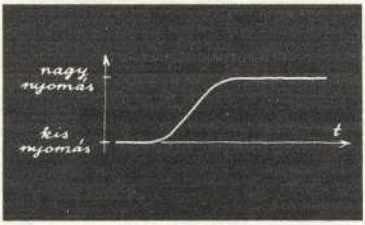
A számológép egyes egységei mind kívánalmak kielégítésére szolgáló operátorok. Hasznos előgyakorlat, ha az Olvasó sorra veszi az eddig megismerteket, és megfogalmaz kívánalmakat, amelyek kielégítése ezekre az operátorokra bízható. Célszerű, ha azzal is foglalkozik egy kicsit, hogy az operátorok *hogyan dolgoznak* a kívánalmak kielégítésén, hogyan végzik igénykielégítésre törekvő munkájukat, hogyan látják el ilyen feladataikat.

A digitális automatán belül az egyik legfontosabb tevékenység az információ áramoltatása. A közönséges vezetéket mi — ellentétben az irodalommal — továbbra is ugyanolyan fő alkatrésznek tekintjük, mint például a maximumképzőt, a minimumképzőt és a negálót.

A vezetékek révén — legyen az pneumatikus, hidraulikus, elektromos vagy üvegből készült fényvezeték, vagy akár a rádióhullámokat továbbító térrész — elsősorban információmásolást **kívánunk** megvalósítani. Ezzel az alapfontosságú kívánalommal és megvalósításával foglalkozunk most részletesebben.

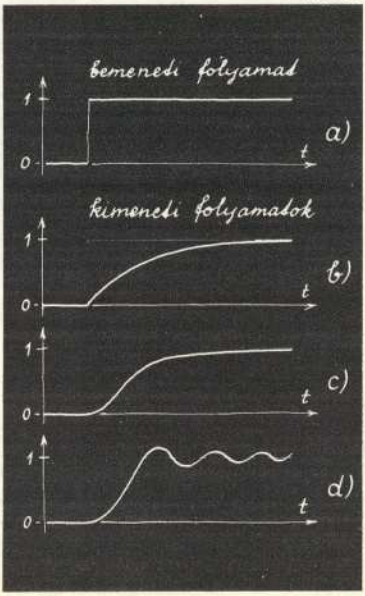


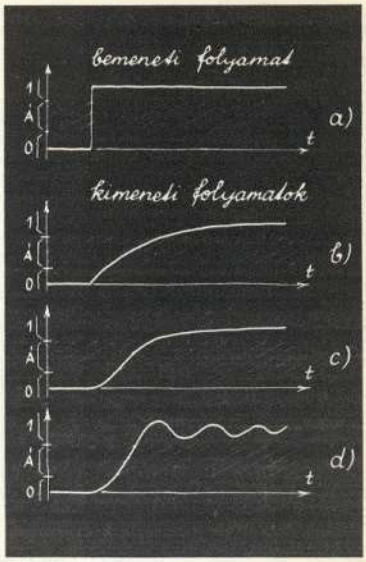
1. ábra



2. ábra

3. ábra





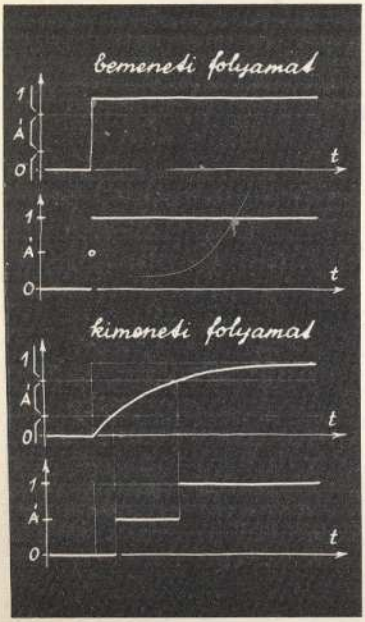
4. ábra

Az információterjesztési kívánalom

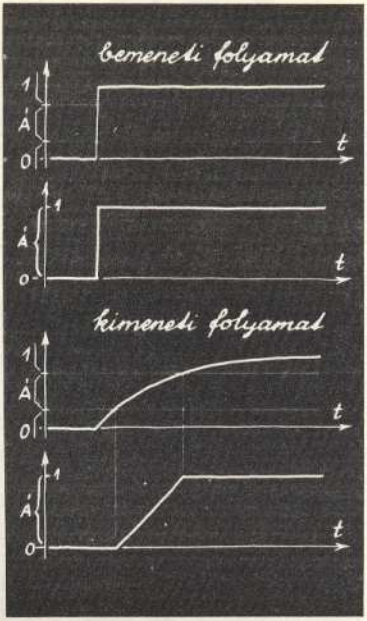
Ha az egyik operátor nem tudja a másiknak továbbadni azt, amit ő tud, és a másiknak is tudnia kell ahhoz, hogy dolgozhasson, a gép nem működőképes. Az információ terjesztésével kapcsolatban tehát nélkülözhetetlen kívánalmaink vannak. Természetesen a puszta terjesztéssel ugyanott lennénk, ahol a terjesztés teljes hiánya esetében volnánk. Gátat is kell szabnunk a terjedésnek. Hiba volna az, ha minden információ mindenhova eljutna és kifejtené hatását. (Rossz az a rádió, amellyel sehol sem lehet semmilyen adást fogni, és az is rossz volna, ha mindenütt minden állomás egyszerre szólna.)

A kívánalmak tehát az információ irányított (*vezetett*, nem pedig szabadjára eresztett!) terjedésével kapcsolatosak. A vezetett információ terjedésére vonatkozó kívánalmakat a mai gépekben főleg vezetékekkel elégítettjük ki.

Mind a tervezés, mind pedig a működés megértése szempontjából nélkülözhetetlen tehát alaposan ismernünk azt, hogy hogyan működnek a vezetékek.

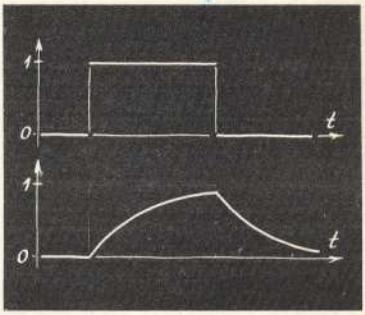


5. ábra



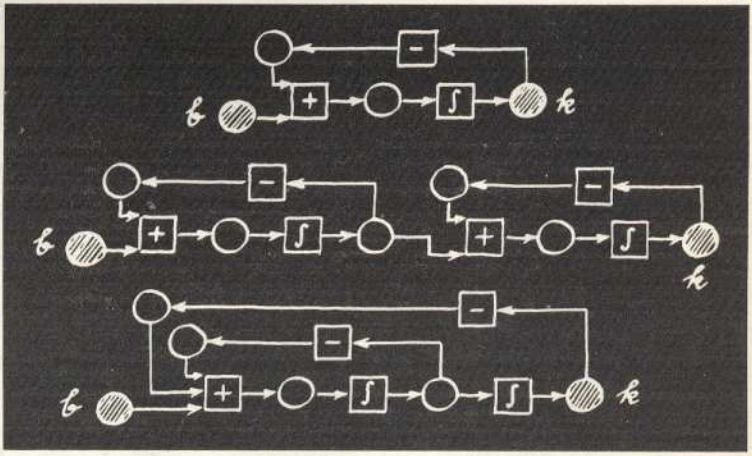
6. ábra

viszont *pontosan* teljesíthető. A *3. ábrán* például a bemeneti eseményből (jelenségből) származóan olyanok jelentkeznek a kimeneteken, amik látszólag nem is hasonlítanak rá. A „szép" derékszögű egyenes szakaszokból olyan görbe vonalak lettek, amik önmagukban szépek ugyan, de az eredetivel egybevetve *sokkal inkább karikatúrák, mint élethű képmások*. És ezzel helyben is vagyunk. Olyan a digitális technika, mint a jó *karikatúrarajzolás*. A jó karikatúrában a lényegesnek tartott vonások megmaradnak (esetleg hangsúlyozva, sőt túlhangsúlyozva is), a lényegtelenek pedig vagy megmaradnak vagy nem.



7. ábra

8. ábra



Első megállapításunk az, hogy nincs egyszerűen működő vezeték. Még a közönséges cső, egy pneumatikus vezeték is annyira gazdag és érdekes rendszer, hogy ha csak a legfontosabbakra szorítkozunk vele kapcsolatban, akkor is meg lehet tölteni egy könyvet csupa érdekes és szép tudnivalóval. Fegyelmezve ismeretszerzési vágyunkat *(kívánalmunkat!)* csak a számunkra közvetlenül fontos dolgokkal foglalkozunk.

Mit kívánunk egy vezetéktől? Természetesen azt, hogy valamit valahonnan valahová elvezessen, *elkalauzoljon*.

Azt kívánjuk például, hogy az egyik végén jelentkező állapot jelentkezzen a másik végén is; jusson el a másik végére is, ami az egyik végén volt. Konkrétan, például az *1. ábrán* látható bemeneti nyomásváltozási jelenség tapasztalható legyen a kimeneten is.

Olyan vezeték azonban, amely ezt a kívánalmat ki tudja elégíteni, nincs. Fontos tudatosítani, hogy ennek oka nem az, hogy az 1. ábrán szereplő folyamat maga sem megvalósítható. Bármilyen „enyhébb" változás *(2. ábra)* pontos továbbítására, kimeneten való utánzására sem találnánk vezetéket.

Kezdjünk el kísérletezni! Egyszerűség kedvéért az 1. ábrán látható hirtelen változással. A kívánalom ennek a bemeneti jelenségnek **utánzása, lemásolása** a kimeneten. A vezeték kimeneti pontja állapotának tehát **követnie** kell a bemeneti pont állapotát. A vezetéknek tehát másoló, utánzó operátornak kell lennie, *követési, másolási, utánzási műveleteket* kell végeznie.

Kísérleteink meggyőznek arról, hogy pontos követő (másoló, utánzó) operátor nincs. Nagypontosságú mérések a legegyszerűbb esetekben a 3. ábrán látható utánzatokat (meglehetősen pontatlan) másolatokat szolgáltatták.

Bele kell nyugodnunk tehát, hogy alapfontosságú operátoraink egyike a másoló, a követő, az utánzó operátor nem tudja pontosan ellátni feladatát. A másolási, utánzási, követő fő feladat pontos megoldása tehát lehetetlen. Azaz az elméletben nagy hasznosságú késleltető operátor pontos megvalósítása a gyakorlatban megoldhatatlan. Ez a tehetetlenség azonban nem végzetes. A digitális módszer lényege éppen az, hogy valaminek valami feleljen meg; nem az a lényeges, hogy mi, csak az, hogy különbözőknek különböző. Ez a kívánalom

Keressük meg, hogy e karikatúrává torzult másolatokban mi az a lényeges, ami megmaradt!

A *4. ábra* ugyanaz, mint a 3. ábra, de egy *„átmeneti állapot sáv"* is szerepel benne. Ennek segítségével a másoló *(valójában karikatúrarajzoló) operátor* tevékenysége a következőképpen írható le. A bemeneti jelenség (esemény) az, hogy a rendszerjellemző az *Á átmeneti sáv* alól föléje kerül. Erről az *eseményről* tudósít az operátor a kimenetén azáltal, hogy a kimeneti rendszerjellemző értéke *is* az átmeneti sáv alól afölé kerül. Hogy mennyi idő múlva, és hogyan, az más kérdés.

Finomabb vizsgálatok

Ha az átmeneti sáv alatti részt a 0 számmal („nem fúj"; kicsi a nyomás, kicsi a feszültség), a felette levőt pedig az 1 számmal („fúj", nagy a nyomás, nagy a feszültség) jelöljük, akkor rendszerünket (legalább) 3 állapotot tartalmazó állapottérben könnyen ábrázolhatjuk. Így, ha az idő előrehaladásának módjával nem törődünk, a másoló operátor tökéletes követési igények (kívánalmak) kielégítésére képes. Ez azonban csak a digitalizált világban érvényes. Ennek lényege — mint láttuk — az állapotok olyan, durva nagy csoportokba foglalása, amelyeknek megkülönböztetése primitív eszközökkel is könnyen elvégezhető.

Azokat az eseteket pedig, amelyek problémát okozhatnak, egyszerűen elkerüljük és elkerültetjük! Esetünkben az átmeneti sávban tartózkodás állapota jut erre a sorsra. Ne tévesszen meg senkit az, hogy például az átmeneti sáv és az alatta levő tartománynak határa kritikus vonal lehet, olyan értelemben, hogy nehézséget okozhat annak eldöntése, hogy egy pont a határvonal felett van kicsivel, rajta van, vagy alatta van kicsivel.

Ez a kérdés szintén nem érdekes, például azért sem, mert a digitális rendszereket úgy szerkesztjük, hogy az ilyen bizonytalanságok ne okozhassanak problémát. Ennek egy lehetséges módja, hogy csak a biztosan eldönthető eseteket számítjuk a 0 vagy az 1 tartományba, a bizonytalanok az átmenetibe kerülnek. Nyilvánvaló, hogy nem fordulhat elő bizonytalanság a 0 és az 1 határán, mert ez a határ nem létezik. És ez a fontos. Ha egy érték bizonytalan, hogy 0 vagy átmeneti, akkor az biztosan nem 1. Ha bizonytalan, hogy átmeneti vagy 1, akkor az biztosan nem 0.

Általános elv az, hogy *mindig csak azzal és csak akkor foglalkozunk, amikor valami biztos és egyértelmű, addig pedig amíg nincs így, vagy várunk, vagy dolgozunk az ilyen helyzetek létrehozásán.*

Tipikus megoldás még a bizonytalan helyzetek kialakulásának eleve lehetetlenné tétele. Így például a rendszert úgy alkotják meg, hogy a 0→1 és az 1→0 átmenet csak gyorsan, meredeken történhessen. Így eleve kizárt már az átmeneti sáv határai közelében való hosszabb tartózkodás is, a határon való „szédelgésről" nem is beszélve. (Az olyan gépeket pedig, amelyekben ez mégis előfordul, javításba veszik.) Olyan ez a kizárásról gondoskodás, mint amikor egy patak átugrásához eleve megköveteljük például, hogy a merőlegestől legfeljebb 10 fokos eltérésű egyenes pályán fussunk neki, és a patak szélétől fél méterre legalább egy adott sebességgel mozogjunk. Ha ezt betartjuk, biztos, hogy rövidesen a túlparton leszünk.

Az *5. ábrán* az iménti elvek szerint állapottérben ábrázoltuk a másolási (karikatúrarajzolási) folyamatot a 4b. ábra esetében. A bemeneti folyamat jellegében nem változott, a kimeneti azonban igen.

A *6. ábrán* megismételtük a 4b. ábra átfogalmazását, de a 0 és 1 állapotban levést az időtengellyel párhuzamos szakasz, az átmeneti részben levést pedig az időtengellyel nem zérus szöget bezáró szakasz képviseli. (Lásd Alapozás XV!) A *7. ábrán* egy impulzus „karikatúráját" mutatjuk be a 4b. ábrát adó vezeték esetében.

Befejezésül közlünk egy-egy vezetékmodellt, azaz karikatúrarajzoló operátorcsapatot. A *8. ábrán* olyan rendszerek kapcsolási rajzai szerepelnek, amelyeknek bemenetére adva az 1. ábrán látható folyamatot, azok arról rendre a 3b., 3c. és 3d. ábrán látható „másolatot" szolgáltatják.

Annak érdekében, hogy megbízható gépeket szerkeszthessünk, a „karikatúrák" legfontosabb tulajdonságait kell még feltárnunk. Ehhez a munkához azonban már számológépet is igénybe vehetünk. Ezért a következőkben vezetékekre vonatkozó rendszermodellekkel foglalkozunk majd; ezek egyben legegyszerűbb képviselői lesznek a számológéppel egyáltalán elvégezhető legbonyolultabb feladattípusnak, az előző értékekre támaszkodó folyamatok, a rekurzív folyamatok értékei kiszámításának is. Ennek sikeres elvégzéséhez azonban már semmi különleges kívánalomnak nem kell eleget tennünk, mert minden lényeges eszköz a kezünkben van már.

POGÁNY CSABA

PC -t a 2C -ből

A NOVOTRADE RT. 1985 decemberében új szolgáltatással bővítette kínálatát

A legalacsonyabb lízingdíjjal IBM PC kompatíbilis Commodore PC 20 számítógépet kínálunk Önnek!

A Commodore PC 20 számítógép-konfiguráció főbb jellemzői:
— Intel 8060 mikroprocesszor, 4,7 MHz
— 256 kbájtos központi memória
— monokromatikus displayegység
— 1 db 360 kbájtos floppyegység
— 1 db 10 Mbájtos Winchester lemezegység

A számítógép teljes lízingdíja 688 000,— Ft, melyet az alábbi részletekben kell fizetni:

a szerződés aláírását követően	a teljes lízingdíj
8 napon belül	50%-a
12. hónapban	20%-a
24. hónapban	15%-a
36. hónapban	15%-a

Szállítás azonnal!

Gazdag szoftverválaszték!

Keresse fel a NOVOTRADE

2C Áruházát! Cím: Budapest XIII., Balzac u. 35.
Telefon: 402-954. Telex: 22-7673.

A jó vezetés titka

Neil Spooney menedzsertanfolyamot tartott Magyarországon. Erről a nálunk még eléggé szokatlan képzésről kérdezem őt.

E. L.: Mindenekelőtt az érdekelne, hogy megtanulható-e a jó menedzserség, elsajátíthatja-e valaki a vezetői készségeket, vagy sem?

N. S.: Természetesen nem tehetünk senkit tanítással jó menedzserré, legfeljebb tudatosíthatjuk benne, hogy milyen a jó menedzser. Megértethetjük vele e tevékenység lényegét, célját, de azután személyes képességein múlik, mire megy.

E. L.: Ön szerint mi ennek a tevékenységnek a lényege?

N. S.: Bizonyos célok elérése, és az, hogy a vezető úgy érje el a céljait, hogy az elfogadható legyen a társadalom, az emberek és annak a környezetnek a számára, amelyben tevékenykedik

E. L.: Tudomásom szerint ön ezeket az ismereteket elsősorban számítástechnikusok körében terjeszti. Valószínűleg ezen a viszonylag új területen nagyon sok hibát követnek el az irányítók, a menedzserek.

N. S.: Hát bizony sok hibát látunk. Én húsz éve dolgozom a számítástechnikai szakmában, elsősorban szolgáltatásokban. Itt az a probléma, hogy az átlagosnál sokkal változóbb területtel van dolgunk. Ilyen helyzetben akkor lehet sikeres az ember, ha hisz abban, hogy meg tudja oldani a problémákat, amelyek a következő héten vagy a következő években felmerülnek. Persze az ilyen önbizalom azon is múlik, hogy tudja-e az ember, hogyan kell irányítani.

E. L.: Előzetes beszélgetésünkben két nagyon fontos szempontot említett ezzel kapcsolatban. Az egyik, hogy a folyamat egészét kell tudni áttekinteni, és nem szabad csupán az egyes részleteket figyelembe venni. A másik, hogy mindent úgy kell nézni, mint valamilyen termelő ipart, amelynek elő kell állítania valamit. És hogy ez menynyire sikerül, abból ítélhetjük meg, hogy sikeres-e az irányítás vagy sem.

N. S.: A legtöbb vezető a vezetést leegyszerűsítve, puszta problémamegoldásnak tekinti. De ha mondjuk egy olyan összetett dolgot kell megoldani, mint a számítógépek bevezetése az oktatásba vagy ennek a technikának elterjesztése a társadalomban, ezt nem nézhetjük túlságosan szűk látószöggel, mert akkor nagyon sok hibát fogunk elkövetni. Például az oktatást úgy kell tekintenünk, mint oktatási ipart, és meg kell vizsgálnunk, melyek azok a nevelési „szerszámok", amelyekkel a legjobb minőségű „terméket" tudjuk előállítani, ebben az esetben a magasan képzett szakembereket. Ha ezt nem üzletként, nem iparként fogjuk fel, akkor nem találjuk meg a megfelelő eszközöket.

E. L.: Bizonyára ennek a szélesebb áttekintésnek egy része az is, hogy valamiképpen mérni kell az eredményességet.

N. S.: Természetesen. Az oktatásban ez meglehetősen nehéz feladat. De azért vannak mértékek (például a vizsgán átjutók száma, a népesség magasabb fokú oktatásban részesülő rétegének aránya stb.), csak éppen mások, mint amilyeneket az üzleti életben megszoktunk. Ki kell tehát dolgozni ezeket, és segítségükkel mérni a hatékonyságot.

E. L.: Nemcsak az oktatásban, hanem például a szoftverkészítésben is sokan követnek el hibát. Például a programkészítők csodálatos programokat készítenek, amelyek esetleg senkinek sem kellenek.

N. S.: Mi, számítástechnikával foglalkozók valóban nemtörődömök voltunk olyan értelemben, hogy nem vettük tudomásul, hogy a valóságos világ részére kell termékeinket előállítani. A termékeknek a piacra juttatása okozza a nehézséget. És még nehezebb ezeket a termékeket eljuttatni olyan piacokra, amelyeken szokatlanok, például a harmadik világba. Hajlamosak vagyunk csak a saját piacainkra termelni, abban a hitben, hogy majd átvihetők termékeink máshová is. De nagyon valószínű, hogy ebben sincs igazunk.

E. L.: Hogyan kerülhetők el ezek a buktatók, honnan tudjuk, melyik a helyes út?

N. S.: Itt visszajutottunk a menedzselés kérdéséhez. Nincs még egy olyan iparág, ahol kifejlesztenek egy terméket úgy, hogy előtte nem elemezték gondosan az igényeket, a felvevőpiacot. Csak a számítástechnikában fordulhat elő, hogy valaki ír egy új programot, anélkül, hogy a vezető kérte volna: bizonyítsa be, hogy erre szükség van. Meg kell tehát teremteni a számítástechnikában is a számításipart. Nézze meg például a mikroszámítógépeket. Miféle programok léteznek ezekhez a nagyon elterjedt masinákhoz? Könyvelési programok, amelyeket kisvállalatok használnak – ez rendben van. De azután? Word-processorok, adatbázis-kezelés, táblázatok, grafika. Régóta ez van, semmi új nem születik. Márpedig kellene új dolgokat fejlesztenünk, különben állóvíz alakulhat ki körülöttünk. Persze elképzelhető, hogy még jobban el kell terjedniük ezeknek a gépeknek, mielőtt felmerülne újabb alkalmazások igénye. Akárcsak a telefonnál: ha ezer emberre jut egy telefon, akkor nemigen fordulhat elő, hogy háziasszonyok egymással csevegnek. Ha viszont tíz emberre jut egy készülék, akkor háziasszonyok tömege fog telefonon társalogni.

E. L.: Beszéljünk egy kicsit általában a vezetésről. Melyek az ön véleménye szerint a legfontosabb szabályai?

N. S.: Nincsenek szabályok. Szerintem a legfontosabb, hogy a vezető higgyen a saját józan eszében. Értsen ahhoz, hogyan motiválja az embereit.

E. L.: De megtanulható-e, hogy hogyan motiváljunk valakit?

N. S.: Természetesen, ennek megvannak az alapszabályai. Ezeket persze alkalmazhatjuk rosszul is, de a technika létezik. A motiváció alapelveihez tartozik például, hogy ha valakinek nem elégítjük ki az alapvető szükségleteit, akkor hiába próbáljuk magasabb szinten motiválni, semmilyen hatást nem érünk el. Ha valakinek nincs mit ennie, nincs ruhája, akkor először ezeket a szükségleteit kell kielégíteni, és csak azután kell társadalmi helyzetét javítani, majd segíteni abban, hogy jobban kibontakoztathassa képességeit. Szóval vannak módszerek, melyek megtanulhatók, de sajnos a legtöbb ember azt a fáradságot sem veszi, hogy elolvassa az ezekről szóló könyveket.

E. L.: Az ön véleménye szerint tehát a jó vezető mindenekelőtt megtanulja a vezetés technikáját.

N. S.: Pontosan. Olyan ez, mint a nyelvtanulás. Nehéz úgy megtanulni egy nyelvet, hogy nem ismerkedünk meg a nyelvtannal. Persze ha megtanultuk a nyelvtant, az nem jelenti azt, hogy jól beszéljük a nyelvet, de legalább megvan az esélye, hogy megértenek bennünket. Ha a vezetésben nem tanuljuk meg, hogy egyáltalán mi az alapja az egésznek, akkor nagyon nehéz jó vezetővé válni.

E. L.: Ön ezalatt a rövid idő alatt bizonyára nem a részleteket tanítja, hanem inkább felhívja a figyelmet arra, hogyan kell mindezt elsajátítani.

N. S.: Három nap alatt annyit érhetek el, hogy ráébrednek, mi mindent kell megtanulniuk; kétségeik legyenek, de ugyanakkor több önbizalmuk is. Higgyenek abban, hogy képesek jó döntéseket hozni. Szeretném felkelteni az érdeklődésüket, hogy elmenjenek és bizonyos dolgoknak könyvtárban, másutt utánanézzenek. Nem kell megtanulniuk, de legközelebb, ha az adott kérdéssel találkoznak, jusson eszükbe: igen, már hallottam erről. És végül meggyőződésem, hogy egy vezetőnek rendszeresnek kell lennie az ismeretek felújításában, a vezetői ismeretek felfrissítésében.

E. L.: Van egy nehézsége a vezetői ismeretek terjesztésének: az egyes kultúrákban ezek eltérőek lehetnek. Ami érvényes Angliában, esetleg nem érvényes Magyarországon.

N. S.: Ez kétségtelenül nehézséget jelent. De elég sokszor megfordultam már Magyarországon, és az a tapasztalatom, hogy az itteni gondolkodás meglehetősen közel áll a modern üzleti gondolkodáshoz. Talán több önbizalom kellene ennek felismeréséhez.

EGYED LÁSZLÓ

Gyors importra van szükség!

Rényi Gábor a Novotrade 36 éves ügyvezető igazgatója. A Szovjetunióban fejezte be felsőfokú tanulmányait, gépgyártás-szervezés-gazdaságtan szakon, 1972-ben. Diplomamunkáját a harkovi traktorgyár szervezéséről írta. Kettős végzettségét elsősorban kereskedelmi területeken hasznosította; a Novotrade-et megelőzően az építőiparban, a gépiparban, majd a Skála kooperációs igazgatójaként a belkereskedelemben.

— Mennyiben tartja magát számítástechnikai szakembernek?

— Bár saját céljaimra tudom használni a számítógépet, nem vagyok számítástechnikai szakember. A szakmai részt persze nagyon fontosnak tartom, ezért vettem magam körül olyan szakemberekkel, akik jól értenek a számítógéphez, programozáshoz, szoftverkészítéshez. A számítástechnika ma már olyan mértékben fogyasztási cikk, hogy úgy vélem, terjesztésében a kereskedők talán többet tehetnek, mint a szigorú értelemben vett szakma. Mert a szakma egy bizonyos körön belül mozog, a kereskedők viszont széles tömegekkel kerülnek kapcsolatba. Én elsősorban kereskedőnek tartom magam, és a Novotrade-nél is egyfajta kereskedelmi munkát honosítottam meg és irányítok.

— Ehhez olyan módszerek is tartoznak, mint amit például az 1985-ös BNV idején alkalmaztak, amikor a vásár helyszínétől távol rendezett bemutatót a cég, a felhasználókat egy dunai hajóra csalogatva? Egyesek véleménye szerint ez költséges rongyrázás.

— A kérdés ilyen megfogalmazásával nem értek egyet, formailag sem, tartalmilag meg különösen nem. Először is a BNV szervezőinél a jelek szerint a hagyományos számítástechnikai vállalatok részesülnek előnyben az újonnan fellépőkkel szemben, akik nem jutnak megfelelő helyhez bemutatkozásukhoz. Másodszor úgy gondoltuk, hogy a Dunán sokkal kellemesebb körülmények között mutathatjuk be kínálatunkat, mint a vásáron, alkalmatlan helyen, millió ember közt tipródva, kulturálatlan körülmények között. Harmadszor: nem éppen az rongyrázás, ha kiszórunk több millió forintot különösebb eredmény nélkül? Mi a hatékonyságot tartjuk a leglényegesebbnek, és miután a számítástechnikát fogyasztási cikként forgalmazzuk, ezeket a szempontokat figyelembe véve is reklámoztuk termékeinket.

— Nem túl merész dolog a Novotrade részéről a számítástechnikai „establishmentnek" szinte kesztyűt dobva — tulajdonképpen kívülállóként — belevetnie magát egy olyan technikába, amely alapos szakmai felkészültséget és megfelelő körültekintést igényel?

— Éppen ellenkezőleg. Ha megnézzük, hogy mi történt a világban az elmúlt néhány évben, látnunk kell, hogy jóformán csak a számítástechnikai „establishmenten" kívül állók szálltak be a szakmába: lemezgyárak, könyvkiadók, fotóoptikai cégek, áruházi láncolatok stb. Mi semmi mást nem tettünk, mint odafigyeltünk erre, és ennek megfelelően cselekedtünk. Igen rövid idő alatt olyan szoftverkínálatot hoztunk létre, amelyet a nagy szakmai felkészültségű vállalatok nem tudtak nyújtani. Mert felismertük, hogy minek van piaca, és kiszolgáltuk ezt a piacot.

— A vevő számára ki garantálja a szoftver minőségét?

— Ebben a kérdésben sikerült előbbre lépnünk. Mindent precízen bevizsgáltunk, és megszüntettük a bizományosi konstrukciót. Körültekintően válogatunk a szoftverek között. A választékot először saját fejlesztőgárdánk nézi át, másodszor a szervezők, harmadszor pedig a külső szakértőkkel lektoráltatjuk. Természetesen fenntartjuk a szerző felelősségét is.

— Mi a véleménye a szoftverkiadó vállalatok kiskereskedelmi tevékenységéről?

— Korszerűtlen és gazdaságtalan, mert nem értenek hozzá. Nem csupán meggyőződésem, hanem tapasztalatom is, hogy ha valaki jó szoftvert tud készíteni, az még nem jelenti azt, hogy el is tudja adni. Miért? Mert nem képes szimplifikálni a vevő számára. Olyan szaknyelven beszél, amit a vevő nem ért, és nem is kell, hogy értsen. A felhasználónak az a dolga, hogy leüljön a gép elé, beadja az adatokat, és válaszoljon a kérdésekre.

— A jelenleg túl magas szoftverárak oka, hogy a teljes választék nem kerül be a hatékony kiskereskedelembe. Az egyedi megrendelésre készített szoftverek még mindig nagyobb üzletnek számítanak.

— Ezek az én filozófiám legnagyobb ellenségei. Magyarországon a számítógépesítés oltárán milliókat elfecsérelve, mindenki a maga rossz, elavult szervezetére akar számítógépes programot, ahelyett, hogy egy korszerű számítógépes programhoz igazítaná a szervezetét!

A jelenleg magas szoftverárak letörésének egyik módja a piaci nyomás fokozása, a kiskereskedelemi hálózat bővítése. Mi például megállapodást kötöttünk a Művelt Nép és az Állami Könyvterjesztő Vállalattal újabb — a C2-höz hasonló — üzletek sorának létrehozására a fővárosban és vidéken egyaránt. Ha hozzáférhető árú szoftvereket tudunk adni a vevőknek, ez egyben segíti a forgalomban lévő számítógépek fajtaszámának csökkenését és darabszámának növekedését. Mi már 250—400 forint között is szállítunk jó minőségű, a világpiacon is versenyképes árú játék- és oktatóprogramokat. A vállalatoknál alkalmazott felhasználói szoftvercsomagok ára nehezebben nyomható le, részben a magát makacsul tartó szemlélet miatt, hogy az olcsó szoftver nem is lehet jó.

Sajnos még mindig nagyon sok a párhuzamos fejlesztés. Azzal nem törődik senki, hogy ez mibe kerül a felhasználóknak, végső soron a népgazdaságnak! A párhuzamos fejlesztések elkerülésére mi egy News Letter kiadásával próbálkozunk, amelyben mindig jelezzük, hogy a Novotrade milyen újabb szerződéseket kötött szoftverek kifejlesztésére.

Egyébként biztos vagyok abban, hogy a háziszámítógéptől a PC-ig bezárólag a magyar szoftveresek abszolút versenyképesek. Csakhogy azzal, hogy nem adunk megfelelő minőségű hardvert a kezükbe, energiájuk nem tőkés deviza termelésére, hanem a belföldi felhasználók kirablására használódik fel. Egyszerűen azért, mert termékeiket külföldi vállalatokéval nem kompatibilis gépekre készítik, így azok ott el sem adhatók.

— És a nem fogyasztói számítástechnika?

— Az tény, hogy óriási szellemi és anyagi erőket elpocsékolva, csillagászati árakon állítunk elő a világpiacon már a gyártás beindulásakor versenyképtelen számítástechnikai termékeket. A hazai gyártás erőltetése és a legfejlettebb színvonalú technika importjának hiánya az egész népgazdaságot visszahúzza. Mindenki azt szajkózza, hogy a tőkés piacon versenyképes árakon kell termelni, miközben a feltételek a számítástechnika területén nem azonosak. Nekem az a véleményem, hogy ha a fejlett technika importjára nem lesz deviza, akkor nemsokára egyáltalán nem fogunk tudni exportálni, mert a feldolgozott termékek részaránya az exportban egyre csökken, és ennek megállítására csak a legfejlettebb technika alkalmazásával van némi remény. Az élenjáró számítástechnika gyors importja központi intézkedést igényel. Fel kell számolni azt a tarthatatlan állapotot, hogy ebben a kérdésben mindenki azt csinál, amit akar és tud! Aki éppen el tudja intézni, hogy hozzájusson importalkatrészekhez, az nekiáll gyártani. Méghozzá drágán, kis szériában, jelentősebb külpiacon eladhatatlanul. Mintha hiányozna a racionális gondolkodás . . .

LACZKA MIKLÓS

Strukturált programtervezés

A sorozat előző cikkében bemutattuk, hogy a program által feldolgozott adatok elemzése hogyan segítheti a program szerkezetének kialakítását. A programszerkezet kialakítása közben előfordulhat, hogy olyan szelekciókat kell alkalmaznunk, amelyek feltételeit nem tudjuk az alkalmazott programnyelven megfogalmazni. Ennek sokszor az az oka, hogy programunk abban az időpontban, amikor a szelekció ágai között választani kellene, még nem rendelkezik a választáshoz szükséges információkkal. Ilyen esetekben a visszalépéses programozás (backtrack) technikáját alkalmazhatjuk.

A visszalépéses programozásnak az a lényege, hogy feltesszük, hogy a szelekció első ágát kell kiválasztanunk, és elkezdjük az adatok feldolgozását ezen feltételezés szerint. A feldolgozás folyamán azonban egy vagy több ponton ellenőrizzük feltételezésünk heyességét, és ha azt tapasztaljuk, hogy feltételezésünk nem volt helyes, ezen az ágon nem folytatjuk tovább a feldolgozást, hanem a másik ág feldolgozását kezdjük el elölről. Közben előfordulhat, hogy az első ágon elkezdett feldolgozásnak olyan következményei keletkeztek — például adatokhoz való hozzáférés elvesztése átolvasás miatt, adatok kiíratása —, amelyeket a másik ág feldolgozása kezdetén meg kell szüntetni. Ezeknek a „nem kívánt mellékhatásoknak" a megszüntetése például úgy történhet, hogy a feldolgozott adatokat az első ágon tömbökben tároljuk, és csak az ág feldolgozásának a végén, amikor már biztosak lehetünk abban, hogy feltételezésünk helyes volt, írjuk ki a kimenő adatokat.

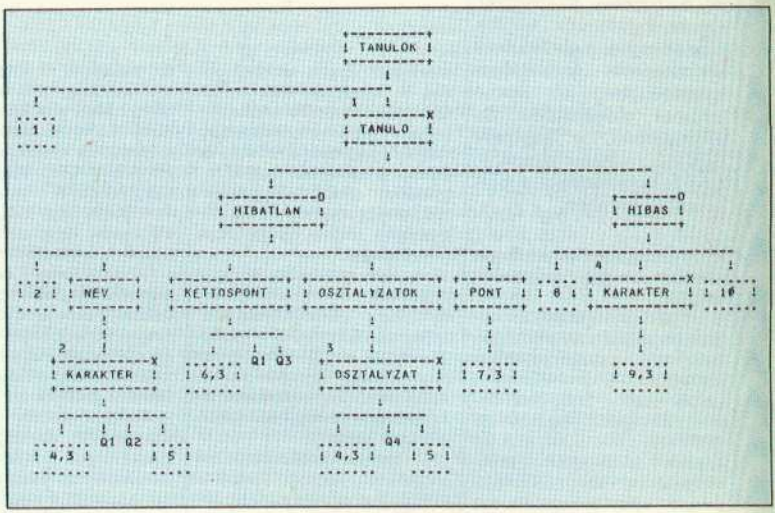
Sok esetben azonban nincsenek ilyen kellemetlen mellékhatások, sőt az első ágon végrehajtott részekre a második ágon is szükség van. Ezek a „kedvező mellékhatások". A „kedvező" jelző arra utal, hogy bizonyos tevékenységek az első ág feldolgozása közben végrehajtódtak, és így ezeket már nem kell a második ágon megismételni.

Azokat a szelekciókat, amelyeket a fenti módon kívánunk feldolgozni, megkülönböztetésül posit-admit szerkezeteknek nevezzük. A „posit" (feltételez) angol szóval az első ágat, az „admit" (megenged, beismer) szóval pedig a második ágat jelöljük.

Amint említettük, a posit-ág feldolgozása közben egyszer vagy többször ellenőrizni kell, hogy jogosan vagyunk-e ezen az ágon. Ennek az ellenőrzésnek a megvalósítására a quit-utasítások szolgálnak (a „quit" angol szó jelentése: hagyd el, fejezd be). A quit-utasítás formája:

quit név *if* feltétel

A quit-utasítás hatása: ha a feltétel igaz, az adott nevű posit-ág feldolgozása ezen a ponton befejeződik, és (ha van) az admit-ág feldolgozása kezdődik el. Ha a feltétel hamis, folytatódik a posit-ág feldolgozása.



1. ábra

Több programnyelv tartalmaz a quit-utasításhoz hasonló utasítást (például a LEAVE utasítás a PL/I-ben, a break utasítás a C-nyelvben). A BASIC nem tartalmaz ilyen utasítást, ezért a quit-utasítás és a posit-admit szerkezet megvalósítása több utasítással történhet.

```
100 REM posit-ág
    .
    .
    .
    IF feltétel THEN GOTO 200
    .
    .
    .
    GOTO 300
200 REM admit-ág
    .
    .
    .
300 REM posit-admit szerkezet
    utáni rész
```

Megjegyezzük még, hogy a posit-admit szerkezetek tetszőleges mélységben egymásba ágyazhatók (ezért kell a quit-utasításban a posit-ág nevét is megadni), és egy posit-ágon több quit-utasítás is szerepelhet.

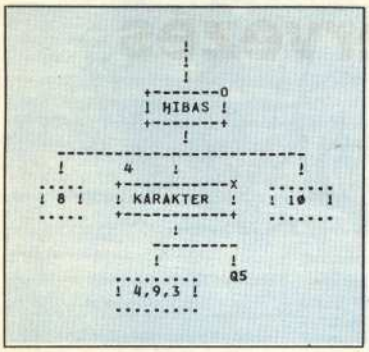
A visszalépéses programozás bemutatásaként az előző cikkben ismertetett 3. feladatnak egy kiegészítését oldjuk meg.

4. feladat. Az x$ karaktervváltozó tanulók neveit és osztályzatait tartalmazza. A nevek az osztályzatoktól kettősponttal vannak elválasztva, és az egy tanulóra vonatkozó részt pont zárja le. Készítsük el azt a programot, amely ellenőrzi, hogy a fenti köve-

```
100 REM tanulók
110 LET i=1:LET y$=""
120 IF i>LEN x$ THEN GOTO 340
130 REM hibátlan
140 LET mi=i:LET z$=""
150 IF x$(i TO i)="." THEN GOTO 200
160 LET c$=x$ (i TO i):LET i=i+1
170 IF i>LEN x$ THEN GOTO 290
180 IF c$<>" " AND
    (c$<"A" OR (c$>"Z" AND c$<"a") OR
    c$>"z")
    THEN GOTO 280
190 LET z$=z$+c$: GOTO 150
200 LET z$=z$+":":LET i=i+1
210 IF i>LEN x$ THEN GOTO 290
220 IF x$ (i TO i)="." THEN GOTO 290
230 IF x$ (i TO i)="." THEN GOTO 280
240 LET c$=x$ (i TO i):LET i=i+1
250 IF c$<"1" OR c$>"5" THEN GOTO 290
260 IF i>LEN x$ THEN GOTO 290
270 LET z$=z$+c$:GOTO 230
280 LET y$=y$+z$+",":LET i=i+1:
    GOTO 120

290 REM hibás
300 LET i=mi:LET t$="" :felt=0
310 IF i>LEN x$ OR felt THEN GOTO 330
320 LET t$=t$+x$
    (i TO i):felt=(x$(i TO i)=" "):
    LET ikie1:GOTO 310
330 PRINT t$:GOTO 120
340 REM y$ feldolgozása
```

2. ábra



3. ábra

```
290 REM hibás
300 LET i=mi:LET t$=""
310 IF i<LEN x$ THEN GOTO 335
320 LET c$=x$(i TO i):LET t$=t$+c$
    :LET i=i+1
330 IF c$<>"." THEN GOTO 310
335 PRINT t$: GOTO 120
```

4. ábra

telmény teljesül-e; ha egy tanulóra vonatkozó rész hibátlan, ezt átírja az y$ karaktervváltozóba, a hibás részeket pedig kiírja a terminálra. A hibás esetek:

a) a név betűn és szóközön kívül más karaktert is tartalmaz

b) az osztályzatok nem 1 és 5 közé eső számok

c) nincs legalább egy osztályzat

d) hiányzik az egy tanulóra vonatkozó részt lezáró pont.

A feladatot megoldó program szerkezetét az *1. ábra* mutatja. A rajzon a Q betű jelzi a quit-utasításokat. Az elemi tevékenységek, feltételek és quit-utasítások a ZX—Spectrum BASIC nyelvén a következők:

Elemi tevékenységek

```
 1 LET i=1:LET y$=""
 2 LET mi=i:LET z$=""
 3 LET i=i+1
 4 LET c$=x$(i TO i)
 5 LET z$=z$+c$
 6 LET z$=z$+":"
 7 LET y$=y$+z$+"."
 8 LET i=mi:LET t$="" :LET felt=0
 9 LET t$=t$+x$(i TO i)
   :felt=(x$(i TO i)=".")
10 PRINT t$
```

Feltételek

1 i=LEN x$
2 x$(i TO i)<>":"
3 x$(i TO i)<>"."
4 i=LEN x$ AND NOT felt

Quit-utasítások

1 *quit* hibátlan if i>LEN x$
2 *quit* hibátlan if c$<>" " AND (c$<"A" OR (c$>"Z" AND c$<"a") OR c$>"z")
3 *quit* hibátlan if x$(i TO i)="."
4 *quit* hibátlan if c$<"1" OR c$>"5"

A program BASIC kódját a *2. ábra* tartalmazza.

Első pillanatra „ügyetlenségnek" tűnhet, hogy a program 220-as és 230-as címkéjű, egymást követő két sora két olyan IF utasítást tartalmaz, amelyben a feltételek megegyeznek. Ezzel kapcsolatban ismét felhívjuk arra a figyelmet, hogy a program BASIC kódját mechanikusan „vezettük le" a programszerkezeti ábrából és az elemi tevékenységek, feltételek és quit-utasítások listáiból. A 220-as címkéjű sor egy quit-utasítást valósít meg, a 230-as címkéjű sor pedig egy iteráció feltételét tartalmazza.

A posit-ágon keletkezett mellékhatások felszámolását a *mi* nevű változó alkalmazásával oldottuk meg. A *mi* változó tartalmazza annak a karakternek a sorszámát, amelynél a posit-ág feldolgozása indul. Az admit-ág feldolgozásának elején az aktuális karakterre mutató *i* változó a *mi* változó értékét kapja meg.

Programunk tervezése folyamán ügyelnünk kellett arra, hogy programunk olyan szövegek feldolgozására is alkalmas legyen, amelyek annyira hibásak, hogy például sem kettőspontot, sem pontot nem tartalmazhat. Programunknak ilyen esetekben is jól kell működnie, azaz nem szabad valamely közbenső utasítás végrehajtása folyamán leállnia értéket nem kapott változó jobb oldalon vagy feltételben való alkalmazására, vagy nem létező láncelemre való hivatkozás miatt (index-túlcsordulás). Ez a magyarázata az admit-ág programjában a *felt* nevű változó alkalmazásának. Az ilyen változók alkalmazása sokszor elkerülhető, ha a quit-utasításokat nemcsak egy posit-ág elhagyására használjuk, hanem egy iteráció végrehajtásának befejezésére is. Erre olyankor lehet szükség, amikor az iteráció magjának végrehajtása kezdetén nem tudjuk egy állításban megfogalmazni a mag végrehajtásának feltételét, mert az ehhez szükséges információ csak az iteráció végrehajtása közben fog rendelkezésünkre állni. A quit-utasításban ilyenkor annak az iterációnak a nevét kell megadni, amelyre az utasítás vonatkozik. A „hibás" nevű rész szerkezetét ilyen quit-utasítás alkalmazásával a *3. ábrán* láthatjuk. Itt az elemi tevékenységek s feltételek a következők:

Elemi tevékenységek

```
 3 LET i=i+1
 4 LET c$=x$(i TO i)
 8 LET i=mi:LET t$=""
 9 LET t$=t$+c$
10 PRINT t$
```

Feltétel

4 i=<LEN x$

Quit-utasítás

5 *quit* karakter *if* c$=":"

A programrész BASIC kódját a *4. ábra* tartalmazza. A kódban a quit-utasítást (330-as címkéjű sor) összevontuk az iteráció végén szereplő GOTO utasítással.

MOLNÁR MÁTÉ

(Folytatjuk)

Magyar–nyugatnémet vegyesvállalat

nagygépes számítástechnikai programozókat keres
teljesítményarányos kereseti lehetőséggel hosszútávú exportmunkára.
Nemcsak egyéni jelentkezők, hanem
vállalatok és társulások jelentkezését is várjuk.

MD Vállalkozási és Kereskedelmi Kft

Budapest I., Döbrentei u. 10. Szabó Sándor nyugatnémet referens
Telefon: 755-327 vagy 755-449

Zöldségeskert

„Semmiképpen sem szabad megengedni, hogy egy új dolog megjelenésével kapcsolatos elnevezések, fogalmak megértése egy szűk szakmai érdekközösség néhány önmagát felszentelt papjának privilégiuma (≙ kiváltsága) legyen egy adott dologhoz való hozzáértése, tudása."

Aki érti, jelentkezzen!

„adat, (ang.: data) különféle dolgok vagy tények, jelenségek többnyire számjegyes (→ numerikus), vagy szöveges (→ alfanumerikus) megadása. Adatfeldolgozáskor adat mindaz, amit µ-SZG számára felismerhető módon lehet tárolni. Az adatok lehetnek bemeneti, kimeneti, törzs, változó vagy mozgó, továbbá numerikus, alfanumerikus és bináris adatok. Röviden: az adat az információ alapeleme, amelyet a µ-SZG feldolgoz vagy előállít."

A heterogén felsorolásból több minden hiányzik, így például az alfabetikus adat, a decimális adat és a logikailag tiszta osztályzás. Mi az, hogy az információ alapeleme?

„adathordozó, a µ-SZG által feldolgozott információt hordozó magnószalag."

A hajlékony lemez mit hordoz? A kitöltött űrlap mit hordoz? Egy nyomtatott könyvoldal mit hordoz? Sajnos van, amelyik csak hibás információt és a felelősség terhét a tudatlan szerző helyett. Javasoljuk, hogy az ilyen nyomdatermékeket hívjuk ezentúl dezinformáció-hordozónak.

„adattömegtár, az a helyiség, szekrény, polc, ahol nagytömegű információt tárolnak. (Hasonlatai: könyvtár, képtár, raktár.)"

Ha egy szekrényben van egy polc, amelyen nagytömegű a tárolt adat, a szekrény vagy a polc vagy mindkettő adattömegtár lesz? Ha pedig a szekrény egy helyiségben van, a helyiség is adattömegtár lesz? Végül pedig, ha e helyiség egy házban van . . .? Csupa eldöntetlen tudományos kérdés. Az azonban biztos, hogy eszerint adattömegtárak karbantartásához asztalos és kőműves szükségeltetik.

„adatvédelem, jogi kérdéskomplexum. Feladata megakadályozni személyi (vagy más) adatok visszaélésre való felhasználását."

A jogi szót olvasva felkészülhettünk, hogy érthetetlen mondat következik. Nem csalódtunk.

„akusztikus csatoló, az az eszköz, amely hallható — például telefonon érkező — hangokat egy számítógép számára érthető digitális jelekké tudja átalakítani. Természetesen az eljárás fordítottját is elvégzi. Helyes elnevezése inkább telefoncsatlakoztató lenne. Angol neve: → modem."

A modem nem azonos az akusztikus csatolóval.

assembler a) alacsony szintű, gépközeli és gépspecifikált programnyelv, amely mnemoniknak nevezett emlékeztető szótöredéket használ, hogy az ember számára is érthető — az elírt feladatra utaló — szavakkal jelöljék az utasításokat a számítógép részére. A mnemonikák közvetlenül fordíthatók bináris kódolású olyan utasításokká, amelyeket a számítógép megért. Használata a magas szintű programozási nyelveknél gazdaságosabb, de a kezdők számára nehezebb, mert a → program is hosszabbá válik. b) fordítóprogram, amely az → egy más programnyelven írt programot gépi kódba tesz át."

Az assembly nem azonos az assemblerrel. Az assembly nyelv és a mnemonikus kódok használata egymástól független. Az „elírt feladat" vajon minek lehet az elírása? Az assembler mint fordítóprogram csak assembly nyelven írt programot fordít, mást nem.

µ-lexikon a µ-számítógéphez (új rovat az Új Technikában, 85/4. szám. Műszaki Könyvkiadó).

A 30 címszó értelmezésének több mint a fele hibás, és úgy fejeződik be a µ-lexikon, hogy „Folytatása a következő kötetben". Mi azonban reméljük, hogy nekünk nem kell majd folytatnunk a szakdudvák és szakkórók gyomlálását a következő Zöldségeskertben.

■

„Bemeneti egység adatok számítógépbe való juttatására szolgáló berendezés. Leggyakrabban billentyűzet, de ~ként működhet a háttértár is."

Ezek szerint a bemeneti egység nem része a számítógépnek.

„Bit egyetlen bináris jegy, amely 0 vagy 1 értéket vehet fel."

Egy jegy hogyan vesz fel értékeket?

„Byte (bájt) nyolcbites kód."

„Kód megállapodás szerinti jelek rendszere."

Eszerint a bájt nyolcbites megállapodás szerinti jelek rendszere lenne. Nem az.

„Fordítóprogram valamely forrásnyelven leírt programból gépi kódú programot készítő program."

A fordítás nincs a gépi kódhoz kötve. Lehet „magas szintű" nyelvről más magas szintű nyelvre is fordítani. Sőt, a fordítás nemcsak nyelvek között lehetséges.

„Forrásnyelv valamely program ember által történő megírásához készült programnyelv."

Tévedés. Forrásnyelv az, amelyről a fordítás történik. Lehet akár gépi kód is.

„Kifejezés változók közötti műveleteket leíró képlet."

Eszerint például változó önmagában nem lehet kifejezés?

„Kulcsszó programnyelvi utasítás, »kódolás«, azonosító része."

Azonosító része van a szómagyarázatnak, érthető része nincs.

„Mikroszámítógép a legegyszerűbb felépítésű, többnyire egyetlen mikroprocesszort tartalmazó számítógép, amely többnyire pusztán gépi kódban programozható."

Élénk képzelőtehetség és teljes függetlenség a szakirodalomtól.

„Operatív memória a működő program számára szolgáló tárolóberendezés a számítógépben."

Nem működő programok és adatállományok tárolása esetén a gép kicsapja a biztosítékot?

„Változó adat tárolására kijelölt memóriarekesz vagy memóriarekeszek."

Ezek is tekinthetők változónak. Tévedés azonban azt hinni, hogy csak ez lehetséges. Ez a változó tárrekeszként van definiálva. Egy matematikai modell változóiból azonban többet is tárolhatunk egy rekeszben, és egy változót is tárolhatunk több rekeszben, mégpedig úgy, hogy egyszer az egyikben, máskor pedig a másikban foglal helyet annak éppen tárolni kívánt értéke.

Ezek és más, ezekhez hasonló pontatlanságok vannak abban a kötetben, amelyet „Barátunk, a számítógép" címen 63 000 példányban adott ki a Móra Ferenc Könyvkiadó, gyerekek számára. Ki védi meg a gyerekeket — akik szakmai önvédelemre nem képesek, tehát védtelenek — az olyasféle ártalmaktól, amikben e könyv olvasása során részesülnek?

Végezetül pedig egy örökzöldség a µ-lexikon a µ-számítógéphez szómagyarázataiból:

„akkumulátor, az a gyűjtő — amelyet regiszternek is neveznek —, a µ-SZG-ben az a hely, ahová ideiglenesen egy szám — pl. egy műveletnek az eredménye vagy egy tárrekesznek a tartalma — betölthető, áttölthető és amelynek tartalmához egy másik szám hozzáadható, abból kivonható vagy azzal összehasonlítható."

Ehhez hasonlóval mintha már találkoztunk volna a „szakirodalomban".

A MINŐSÉGÜGY KÖZÜGY

A számítástechnika szerepe a minőségügyi munkában

A minőségnek és a számítástechnikának a kapcsolata legalábbis kettős. Lehet és kell is foglalkozni a számítástechnikai eszközök és tevékenység minőségével, és lehet is és kell is alkalmazni a számítástechnikai eszközöket a minőséggel kapcsolatos minden érdemi tevékenységben.

Mivel mindkét terület különlegesen nagy fontosságú, igéretünkhöz híven foglalkozunk is velük, törekedve, hogy saját ez irányú tevékenységünk minőségét a téma fontosságának megfelelően magas színvonalon tartsuk. Ennek érdekében mindenekelőtt el kívánjuk kerülni a minőségromlás leggyakoribb okát, a mohóságot. „Aki sokat markol, keveset fog!" — mondja a nép bölcsessége. És ugyanez az időre vonatkoztatva : „Lassan járj, tovább érsz!"

A számítástechnika alkalmazása szellemi munka. A szellemi munka minőségének pedig egyik meghatározó erejű jellemzője a fogalmak tisztázottsága. Nyilvánvaló, hogy nem lehet feladatunk filozófiai nehézségű fogalomtisztázás. Ennek két oka is van. Először is sem a minőségügy, sem a számítástechnika tudománya ma még nem eléggé érett ahhoz, hogy ilyen tevékenység haszonnal elvégezhető lenne velük kapcsolatban. Másodszor pedig esetünkben olyan gyakorlati tudományokról van szó, amelyeknek gyorsan fejlődő és gyorsan avuló részei vannak, így nem éri meg olyasvalamin (és ennek alkalmazásán) túl sokat elmélkedni, amit esetleg már holnap egy egészen más valami fog végérvényesen felváltani.

A tisztaság és a tisztázottság azonban nem a filozófia és nem az axiomatikus tárgyalás kiváltsága, erénye az minden jól használható gyakorlati tudománynak is.

A fogalmak tisztázása tehát elengedhetetlen minőségi követelmény. Ezért foglalkozunk ezzel is, nem filozófiai, hanem gyakorlati igényeket szem előtt tartva.

A CAD (Computer Aided Design) és a CAM (Computer Aided Manufacturing) — amelyeket számológéppel segített vagy támogatott tervezésnek és gyártásnak fordítanak — mellett sok más CA kezdetű szó és neki megfelelő hasznos tevékenység van.

Engedtessék meg, hogy a CA kezdetű rövidítéseket itt ne szó szerint, hanem úgy értelmezzük, hogy *számítástechnikával kiszolgált* vagy számítástechnikát használó, számítástechnikát hasznosító.

Az angol nyelvű eredetiben a gép segít, és a gép támogat. Tehát a gép az aktív, a segítő, a támogató, mi vagyunk a passzívak, segítséget, támogatást elfogadók. Sőt a gép még erkölcsileg is felettünk van, hiszen segít és támogat, mi pedig parazitákként ezt a nemes gesztust egyszerűen csak bezsebeljük.

Durva kísérlet ez a gép mítoszának erősítésére és fétisek kialakítására. Mivel a mítoszok és a fétisek a műszaki tudományok értékének alattomos alásói és a fejlődés nehezítői, tudatosan kell ellenük küzdenünk.

Az SzKM tehát a következőkben számítástechnikával kiszolgált minőségügyi tevékenységet jelent. Akinek nem tetszik, használja az SzHM rövidítést, a számítástechnikát használó vagy hasznosító minőségügyi munkára. Nyilván nemcsak a neveken múlik egy tudomány alkalmazhatósága. A jó nómenklatúra és jelölésrendszer azonban óriási érték. Nem véletlenül maradt fenn egy nagy matematikus intelme, hogy „Ne irgalmazzatok a rossz kifejezéseknek!" — azaz dobjátok ki a rossz szerszámokat!

A minőség fogalmával kell kezdenünk a fogalomtisztázó munkát. A minőség szó ma már rokonértelmű az értékességgel. Nyilvánvaló tehát, hogy jelentésváltozáson ment keresztül. A minőség hajdan a milyenséggel volt azonos értelmű (lásd a *milyen? minő?* kérdőszópárt!). A jelentésváltozást — úgy véljük — az okozhatta, hogy a milyenségről mindig csak akkor esett szó, ha értékelést, értékesség-mérlegelést kellett végezni, azaz csak akkor beszéltek a milyenségről (minőségről), ha a mai értelemben vett minőség volt a fontos.

Megállapíthatjuk tehát, hogy a hajdan azonos „milyenség" és „minőség" ma lényegesen különböző jelentésű, és csak a milyenség tartotta meg eredeti jelentését. Ezért hangzik kicsit furcsán a mennyiségnek a minőségbe való átcsapásáról szóló tétel. Nem inkább a mennyiségnek a milyenségre gyakorolt hatásával kellene foglalkozni?

A minőség a gyakorlat számára valamilyen gyakorlati módon mérhető értékesség-jellemzőt jelent. Nincs szükség az értékesség értelmezésébe bocsátkozni, hiszen erre vonatkozóan valamilyen jól definiált mérési eljárás meglétét követeltük meg. Ha ilyen nincs, akkor létre kell hozni.

A minőség testvérfogalma a megbízhatóság. Megbízhatóságon valamilyen tevékenység (általában szolgáltatórendszer által végzett szolgáltatás, gyakran olyan, amelyet valamilyen géppel végeztetünk) valamilyen célra alkalmazhatóságának, alkalmasságának a minőségét azaz értékességét értjük.

A minőséggel való foglalkozás legalább olyan régi, mint amilyen régi az árutermelés és a piaci verseny. A tudatos tudományos és széles körű minőségügyi tevékenységet azonban a kapitalista árutermelés és verseny hívta életre. A tömegtermelés terén, mivel a matematikai statisztikai eszközök ott jól alkalmazhatónak bizonyultak, hamar magas színvonalat ért el a quality control, azaz a minőségirányítás.

Lényeges itt egy pillanatra megállni. Az angol control szó vezérlést, szabályozást, irányítást és ellenőrzést is jelent. Elég egy rövid megfontolás, hogy itt nem az ellenőrzés, hanem az összes többit magába foglaló irányítás fogalmáról van szó. Miért ellenőrizték ugyanis a minőséget? Nemcsak azért, hogy tudják, hogy az milyen. Az ellenőrzés során szerzett információ vissza is hatott a gyártási folyamatra, tehát létezett és működött egy irányítási rendszer is, nemcsak egyszerűen ellenőriztek valamit. Ha tehát minőségirányításról hallunk, gondoljunk arra, hogy ebben mindaz benne van, amit a „statisztikai minőség-ellenőrzés" címszó alatt tárgyalnak a szakmunkák.

A klasszikus kézi adatfelvételes minőségirányító rendszerekből töretlenül fejlődtek ki azok a számítástechnikára alapozott hatalmas minőségirányító rendszerek, amelyekben minden automatikus már. Nemcsak a gyártás, hanem a gyártmány minőségi mutatóinak a „kézbentartása" is.

Érdekes volna a történelmi fejlődés útját ezeken az oldalakon végigjárni; lépésről lépésre megismerve az egyre fejlettebb rendszereket, befejezve a gyártó és minőségirányító automatákkal (robotokkal). Erről a csemegéről azonban hely- és időhiány miatt le kell mondanunk, hogy minél előbb eljussunk addig, hogy az olvasó hozzáfoghasson egy mikrogépet használó minőségirányító rendszer létrehozásához.

Tudott dolog, hogy a minőségirányítás hőskorában volt olyan tőkés, aki annyira tudatában volt annak, hogy mit ér a gyárában minőségirányítással foglalkozó statisztikus munkája, hogy a kiváló statisztikusnak, Gossetnek csak más néven volt szabad eredményeit publikálnia, mert ha saját néven publikál, a versenytársak rájöttek volna arra, hogy a tőkés egy statisztikust foglalkoztat a gyárában, nyilvánvalóan azért, mert az nyereségét növeli. Ennek következménye, hogy ma is Student (ez az álnév) próbának hívjuk a Gosset által szerkesztett próbát.

Mi a helyzet ma? Talán az, hogy a minőségirányítással foglalkozó statisztikusok egynémelyikének ma is álnevet kellene használnia, de más okból. Azért, mert sok helyen olyan alacsony a minőség nálunk, hogy szégyent hoz a minőségirányító szakmára. Pedig az eszközök és az anyagi feltételek terén sok vonatkozásban toronymagasan állunk a múlt felett. A minőségirányító munkát hajdan keservesen megnehezítő számolás terhét pedig teljesen gépre bízhatjuk. A lelkiismeretességet azonban nem. Nyilvánvalóan van a minőségirányításnak is minősége, és ennek egyik fő meghatározója az emberi munka lelkiismeretessége, az a bizonyos sokat emlegetett „dolgozni, csak pontosan, szépen". Prózai változata pedig: dolgozni csak a jól megtervezett minőségben érdemes. Még pontosabban: *dolgozni a jól megtervezett minőségben legérdemesebb.* Először tehát a minőség megtervezésével kell foglalkoznunk.

A PRINTER BASIC

Ez a program a Commodore 64 BASIC utasításkészletének bővítésére szolgál. Autostart EPROM cartridge formában került forgalomba. Ahogy arra neve is utal, elsődleges célja a C64-hez leggyakrabban kapcsolt grafikus nyomtatók kezelésének megkönnyítése. A programozó olyan lehetőségekhez jut általa, amelyeket egyébként a hagyományos BASIC-utasítások legrafináltabb kombinálásával sem lehet megvalósítani. A programfejlesztés folyamatát bőséges parancskészletével támogatja.

A PRINTER BASIC

a következő típusú
nyomtatók kezelését támogatja:

— Seikhosa GP 100/A — Seikhosa GP 100 VC
— Commodore VC—1515 — Commodore VC—1525 — Commodore MPS—801 — Commodore MPS—803

480 × 392

A gyártó cég meglehetősen primitív BASIC nyelvvel hozza forgalomba Commodore 64 típusú személyi számítógépét. Mindaddig nincs is semmi gond vele, amíg csak játék céljaira használjuk. A gépi kódban készült játékok a végsőkig képesek kihasználni a C64 hardverrepertoárját. Mindjárt másképpen fest azonban a helyzet, ha arra gondolunk, hogy Magyarországon e géptípust döntő többségben műszaki és gazdasági feladatok megoldására állították és állítják munkába. Természetesen ilyenkor már fontos szerepet kap a lemezmeghajtó és a sornyomtató is.

Gyakran merül fel az igény nagy felbontású finomgrafikák készítésére, amelyeket célszerű lehet nemcsak a képernyőn, hanem nyomtatón is megjeleníteni. A C64 képernyője 320 × 200-as méretű finomgrafikák kivetítésére alkalmas, a sornyomtatók grafikus képességei azonban sokkal jobb felbontást is lehetővé tesznek. Aki már próbálta, az tudja: ennek kihasználásakor a programozó súlyos korlátokba ütközik az alap BASIC-et használva. A PRINTER BASIC-ben dolgozva azonban már a kezdők is könnyedén megbirkózhatnak a nyomtatók grafikus üzemmódra kényszerítésével.

Egyszerű, könnyen megjegyezhető BASIC utasítások segítségével 480 × 392-es méretű ábrákat készíthetünk, amelyeket aztán egyetlen utasítással kinyomtathatunk, ablakként használva a képernyőt, részletekben megjeleníthetünk, vagy éppen lemezre is kimenthetünk. A 480 × 392-es méretű ábrák minősége közel jár a C64-nél sokkal drágább IBM PC típusokkal készíthető, 640 × 200-as képekéhez.

Működtetése

A gép kikapcsolása után a cartridge a hátoldalon található EXPANSION PORT-ba illeszthető be. A C64 bekapcsolása után a PRINTER BASIC azonnal működésbe lép, kiírja bejelentkezési üzenetét. A BASIC programok számára 26622 bájt szabad memória áll rendelkezésre. A 480 × 392-es méretű ábrák helyfoglalása miatt tehát csökken a rendelkezésre álló szabad hely mérete. A sokféle előny mellett ez a hátrány elkerülhetetlen. A gép kikapcsolása nélkül a következő utasítással hozhatjuk alaphelyzetbe a PRINTER BASIC-et:

sys 64738

A hatás ugyanaz lesz, mint ha ki-, majd bekapcsoltuk volna az alapgépet.

Memóriaszervezése

A PRINTER BASIC interpretere EPROM-ba égetve a $8000-tól $9fff-ig terjedő címtartományban helyezkedik el. Bizonyos részek csak RAM-ban működőképesek, ezek a rutinok bekapcsolás után átmásolásra kerülnek a $cc00-tól $cfff-ig tartó RAM-ba.

A 480 × 392-es ábra minden pontjának megfelel egy bit a C64 tárában. A printergrafika tártérképe $7000-tól $cbff-ig terjed. Ugyanazt a címtartományt használja, mint a programot tartalmazó EPROM cartridge, és átfedésbe kerül az alap BASIC ROM-jával is, ezáltal szinte teljesen kihasználja a C64 tárkapacitását. A printergrafika részleteire ablakokat nyithatunk a képernyő felhasználásával. A képernyő bittérképét a PRINTER BASIC a $e000-tól $ffff-ig található RAM területre helyezi, átfedésben a Kernal ROM-mal.

A kiterjesztés technikája

Az alap BASIC legfontosabb rutinjai RAM vektorokon keresztül futnak le. A vektorokban található címek megváltoztatásával ezek az interpreter rutinok kibővíthetők, adott esetben teljesen le is cserélhetők. Itt olyan részletekről van szó, amelyek felismerik a BASIC utasításokat, token formára hozzák azokat, a tokeneket listázható karakterekre fordítják le stb. Mondhatjuk tehát, hogy a BASIC interpreter lelkivilágához ezeken a RAM vektorokon keresztül férkőzhetünk közel, amelyek a $0300-tól $030b-ig terjedő memóriaszakaszban foglalnak helyet. A PRINTER BASIC kifejlesztése is e vektorok megváltoztathatóságának köszönhető. Nagy előnye a módszernek, hogy nem lassítja a BASIC-ben írott programok futási sebességét, ellentétben például a gyakran alkalmazott CHARGET beékelési technikával.

BASIC fordítók használata

Mivel a PRINTER BASIC az új utasításokat új tokenek bevezetésével éri el, lényegileg nem változtatja meg a BASIC nyelv alapfilozófiáját. Ennek a szerencsés módszernek köszönhető, hogy a széles körben elterjedt AUSTROCOMP és AUSTROSPEED fordítók problémamentesen képesek lefordítani a PRINTER BASIC-ben írt forrásprogramokat. A lefordított programok futási ideje természetesen lerövidül.

Bővített utasításkészlet

A könnyebb áttekinthetőség kedvéért érdemes két csoportba gyűjteni az utasításokat. Az első csoportba azok a grafikus utasítások kerülnek, amelyek a printergrafikák készítését segítik elő, a másikba pedig azok, amelyek a programozó kényelmét szolgálják. Nem lehet célja a termékismertetőnek az utasítások részletes szintaktikai leírása, már csak a terjedelem szűkös volta miatt sem. A PRINTER BASIC hivatkozási kézikönyvében ezek hiánytalanul megtalálhatók.

Grafikus utasítások

A PRINTER BASIC a 480 × 392-es ábrák kialakításához könnyen kezelhető, egyszerű BASIC utasításokat szolgáltat.

PLOT. Az ábra bármely pontját írhatjuk, törölhetjük vagy invertálhatjuk vele.

LINE. Két pontot egyenes vonallal köthetünk össze, a vonalat az optimális pontkombinációból állítja össze.

PUT. Egy 3×3-as méretű tömött blokkocskát nyomtat a meghatározott helyre. Ír, töröl és invertál, a többi grafikus utasításhoz hasonlóan.

POINT és JOIN. Használatával folytonosan rajzolhatunk görbéket. Függvények ábrázolásánál igen hasznos.

FRAME. Egy pont szélességű vonalakkal téglalap alakú keretet hoz létre.

CIRCLE. Körök, ellipszisek készítésére szolgál. Különlegessége, hogy elforgatott, dőlt ellipszisek rajzolására is lehetőséget ad.

BLOCK. Segítségével egy téglalapnyi területet törölhetünk, besatírozhatunk, invertálhatunk.

MOVE. Az ábra valamely részterületét átmásolhatjuk egy másik helyre.

TEXT. Alkalmazásával az ábra meghatározott részébe tetszés szerinti ASCII szövegeket írhatunk be, nagyítva is.

ROBOUT. A 480×392-es grafikus ábra bittérképének törlésére szolgáló utasítás.

WINDOW. Képernyőre vetíti 320×200-as részletekben a nyomtatóra szánt nagy felbontású grafikát. Grafikus ablaktechnikai kezelést tesz lehetővé BASIC-ből.

COPY. A PRINTER BASIC leglényegesebb, kulcs utasítása. Az elkészült 480×392-es ábrák nyomtatását végzi papírra. Gyors, optimalizált utasítás. Más programokkal ellentétben, nem rángatja a printer írófejét, a redundáns információk kiszűrésével a lehető legsimább fejmozgatással végzi a nyomtatást.

HISAVE. A nagyméretű grafika bittérképének kimentését végezhetjük vele lemezre.

HILOAD. Lemezen tárolt ábrákat tölthetünk be segítségével. Betöltés után a már a tárban levő ábra vagy teljesen felülíródik, vagy az új ábra csak ráíródik a régire.

A programfejlesztést támogató utasítások

AUTO. Programíráskor automatikusan kiíródik a következő sorszám.

DELETE. Programszakaszok gyors törlésére használható parancs.

NUMBER. BASIC programjainkat sorszámozhatjuk át vele. Átszámozás után a helyes értékek lesznek a GOTO, GOSUB, IF . . . THEN, ON . . . GOTO, ON . . . GOSUB, RUN mögött is. Nemcsak a program elejétől lehet kérni az átsorszámozást, hanem részletekben is újraszámozhatjuk programjainkat.

FIND. Megadott programrészletek kikeresését végzi. A megtalált soroknak nemcsak a sorszámát írja ki, hanem a teljes sorról ad listát.

DIR. A gépben levő program kitörlése nélkül, a lemez tartalomjegyzékét jeleníti meg a képernyőn.

DISK. A parancscsatornával való manipulációk nélkül, egyszerűen adhatunk ki általa parancsokat a lemezegységnek.

ERROR. A lemezegység állapotát jelző hibacsatorna aktuális tartalmát jeleníti meg a képernyőn, inverz karakterekkel. Nemcsak programmódban használható!

LLIST. A program listáját közvetlenül a sornyomtatóra küldi. Fájlok megnyitásával, zárásával nem kell a programozónak törődnie.

HARD. Kimásolja a képernyő tartalmát papírra. Az információkat változtatás nélkül viszi ki, még az inverz idézőjelek sem zavarják meg a működését.

MERGE. Ha a tárban van egy főprogramunk, szubrutinokat, programrészeket másolhatunk hozzá lemezről. A beolvasásra kerülő programrészletet a sorszámainak megfelelő helyre teszi. A főprogram bármelyik szakaszába befésülhetjük az új részletet. A lemezről behívott rutin tehát nem egyszerűen a főprogram végéhez csapódik, hanem úgy ékelődik a főprogramba, mintha billentyűzetről gépeltük volna be. Valódi programépítkezésre ad ezáltal lehetőséget.

POOL. A lemezegységet utasítja, hogy egy szekvenciális fájl tartalmát nyomtassa ki sornyomtatóra. A folyamatban a C64 a továbbiakban már nem vesz részt, a kábel kihúzása után le is kapcsolható. Az utasítás nem bizonytalankodik, minden esetben beindítja a listázást.

COLOR. Beállítja a képernyő színeit.

CURSOR. A következő nyomtatási pozíciót a meghatározott helyre állítja. A PRINT vagy INPUT itt írja majd ki az üzeneteket.

PRINT USING. Táblázatok kiírásához nélkülözhetetlen utasítás. Beállítható a nyomtatandó számban a tizedesjegyek száma, a tizedespontot követő megfelelő számú nullát kényszerít ki.

A PRINTER BASIC hasznos eszköz lehet olyan programozók kezében, akik a műszaki-gazdasági életben kívánják hasznosítani a Commodore 64-et. Különösen fontos kezdő programozók számára, mivel bonyolult feladatok megoldását is lehetővé teszi számukra. A grafikus sornyomtatók gazdaságos kihasználásához nélkülözhetetlen.

Forgalmazás, megrendelés, tesztelés, ár

A program ára 7000 és 8000 forint között mozoghat; kisebb eltérések lehetnek az egyes forgalmazók árai között. A PRINTER BASIC azonnal megvásárolható az alábbi számítástechnikai szaküzletekben:

Novotrade RT. 2C Számítástechnikai Áruház, Budapest, XIII., Balzac u. 35.
Zrínyi Miklós Könyvesbolt, Pécs, Jókai u. 25.

A FOTOELEKTRONIK I. SZ. szakboltjaiban:

Budapest, V., Múzeum krt. 19.
Budapest, XIII., Rajk László u. 46/b
Miskolc, Korvin Ottó u. 5.
Debrecen, Szabó I. altb. tér 6.
Győr, Bem tér 1.

A termék megrendelhető a következő címen:

HARDSOFT Számítástechnikai GT. Százhalombatta, Rózsa Ferenc u. 32.

Iskolák itt 50 százalékos árkedvezménnyel rendelhetik meg a PRINTER BASIC-et!

A HARDSOFT GT. szolgáltatása, hogy kérés esetén kipróbálásra, tesztelésre kiküldi postán a programot a referencia-kézikönyvvel együtt az érdeklődőknek. Amennyiben az érdeklődő hasznosnak találja, kifizeti az árat, egyébként postán visszaküldi a forgalmazónak a programot.

COMMODORE 64

Visszaszámlálásos időmérés

A program eredeti változata az 1985 januárjában a Gellért Szállóban a COMPORGAN Rendszerház KV. által rendezett nemzetközi bridzsversenyre készült.

A versenyen páros és csapatversenyeket bonyolítottak le. A páros versenyek és a csapatverseny időméréssel szemben támasztott követelményei eltérnek, de mindkét esetben elsődleges szempont a játékidő nagyméretű számokkal történő kijelzése és a pontos idő kiírása a képernyő jobb felső részére. A program által működtetett órát tévélánc segítségével láthatták a versenyzők és a nézők.

A páros versenyek során beállított versenyidő elteltével 1 perc szünet következett. Eközben a párok helyet cseréltek. A szünet után a beállított játékidőtől automatikusan újra indult a visszaszámlálás.

További követelmény volt, hogy 3, 2 és 1 perc hátralevő játékidőnél a képernyőfelirat szint váltson, és különböző hangjelzések szólaljanak meg.

A csapatversenynél nem volt szükség a visszaszámlálás automatikus újraindítására, viszont a szín- és hangjelzéseknek 5, 3 és 1 perc hátralevő játékidőnél kellett bekövetkezniük.

A bemutatott program ennek az időmérő programnak az egyszerűsített változata, amely csak a játékidő mérésére alkalmas.

Jól használható különböző iskolai vagy egyéb sportversenyeken, vetélkedőkön.

A program kezelése és működési elve

Az óra indítása előtt percekben kell megadni a mérendő játékidőt. Ezután azt az időpontot kell megadni négy numerikus karakterrel — órák, percek —, amikor a versenyt indítani fogjuk. Ezután a program várja a beállított startidőpont beálltát, amit a szóköz lenyomásával kell jeleznünk a gép számára. Ennek hatására a program a beállított játékidőnek megfelelő számjegyeket és a pontos időt felírja a képernyőre.

A másodpercek és a percek változását egy szubrutin figyeli a gép belső órája által működtetett TI$ változó figyelésével. A TI$ egy 6 karakter hosszúságú sztring, amely a pontos időt tartalmazza óra (2 karakter), perc (2 karakter), másodperc (2 karakter) formában.

A programindításkor általunk beállított startidő ebbe a TI$ változóba kerül, és az óra indítása után a számítógép ettől az időponttól kezdve másodpercenként automatikusan lépteti TI$ értékét.

```
1 REM ************************************
3 REM SCHELL FERENC     1986.1.1
4 REM
5 REM ************************************
9 PRINTCHR$(142) PRINTCHR$(8)
10 DIML$(15) K$="■■■■■■■■■■■■■■■■■■■■■■■■"
20 PRINT"■■ ORAMU                                   "
30 PRINTCHR$(142)CHR$(8)
40 O$="■■■■" V=53280
50 POKEV,0 POKEV+1,0 C$="■"
60
70 REM  * JATEKIDO LEKERDEZESE *
75
80 PRINTO$"■■■■■■"
90 INPUT"JATEKIDO PERCBEN [2 KARAKTER!] ";J$
100 J=VAL(J$) IFLEN(J$)<>2ORJ<1ORJ-INT(J)<>0THEN80
110 J=VAL(RIGHT$(J$,1)) L=VAL(LEFT$(J$,1))
120
130 REM * ORA BEALLITASA, INDITASA *
135
140 PRINTK$"                           ...."
150 INPUT"■PONTOS IDO (/ORA PERC/) ";T$
160 PRINTK$" ■ START ■■ -->■■ SPC ■                  "
170 PRINTK$"■■ ■ "+LEFT$(T$,2)+" ORA "+RIGHT$(T$,2)+" PERC "
180 GETA$ IFA$<>CHR$(32)THEN180 REM * <SPACE>: ORA INDUL *
190 TI$=T$+"00" TA$=MID$(TI$,3,2) HA$=TA$
200 PRINTO$"■■■■■■                                  "
210 PRINTK$"■■■                                   ■"
220 PRINTK$"■■          ■J A T E K    I D O■      "
230 PRINTK$"■■                                    ■" GOSUB450
240 GOTO280
250
260 GOSUB390
270
275 REM * HATRALEVO JATEKIDO KIIRASA *
276
280 V=L F=0 TA$=HA$ FORI=1TO15 L$(I)=C$ NEXT
290 ONVGOSUB1100,1200,1300,1400,1500,1600,1700,1800,1900
300 IFV=0THENGOSUB1000
310 FORI=1TO15 L$(I)=L$(I)+" " NEXT
320 ONFGOTO340
330 V=J F=1 GOTO290
340 PRINTQ$ FORI=1TO15 PRINTTAB(9)L$(I) NEXT IFJ=0ANDL=0THEN END
350 IFJ=0ANDL<>0THENL=L-1
360 IFJ=0THENJ=9 GOTO260
370 J=J-1 GOTO260
380
385 REM * MASODPERC ES PERC VALTOZASAT ERZEKELO RUTIN *
386
390 GETA$ IFA$="■"THEN20
400 IFT$<>TI$THENGOSUB450 REM * MP VALTOZAS -> UJ IDO *
410 HA$=MID$(TI$,3,2)
420 IFTA$=HA$THEN390
430 RETURN REM * PERC VALTOZOTT HATRALEVO IDOT KIIRNI (270)
440
445 REM * PONTOS IDOT KIIRASA *
446
450 PR$=LEFT$(TI$,2)+" "+MID$(TI$,3,2)+":"+RIGHT$(TI$,2)
460 PRINT"■■"TAB(19)"PONTOS IDO  "PR$ T$=TI$ RETURN
470
1000 L$(1)=L$(1)+" ■      ■ "
1010 L$(2)=L$(2)+"■        ■"
1015 L$(3)=L$(3)+"■ ■    ■■ ■"
1020 FORI=4TO12 L$(I)=L$(I)+"■ ■      ■ ■" NEXT
1025 L$(13)=L$(13)+"■ ■    ■ ■"
1030 L$(14)=L$(14)+"■■      ■"
1040 L$(15)=L$(15)+" ■■     ■ " RETURN
1050
1100 L$(1)=L$(1)+"     ■ ■  "
1105 L$(2)=L$(2)+"    ■  ■  "
1110 L$(3)=L$(3)+"   ■   ■  "
1115 L$(4)=L$(4)+"  ■ ■■ ■  "
1120 L$(5)=L$(5)+" ■ ■  ■ ■  "
1125 L$(6)=L$(6)+" ■■ ■  ■ ■  "
1130 FORI=7TO15 L$(I)=L$(I)+"      ■ ■  " NEXT RETURN
1190
1200 L$(1)=L$(1)+" ■      ■ "
1205 L$(2)=L$(2)+"■        ■ "
1210 L$(3)=L$(3)+"■ ■    ■■ ■ "
1215 L$(4)=L$(4)+"■■      ■ ■ "
1220 L$(5)=L$(5)+"      ■ ■ "
1225 L$(6)=L$(6)+"     ■ ■ "
1230 L$(7)=L$(7)+"    ■ ■  "
1235 L$(8)=L$(8)+"   ■ ■   "
1240 L$(9)=L$(9)+"  ■ ■    "
1245 L$(10)=L$(10)+" ■ ■      "
1250 L$(11)=L$(11)+" ■ ■      "
1260 FORI=12TO1[illegible] L$(I)=L$(I)+" ■ ■      " NEXT
1265 FORI=14TO15 L$(I)=L$(I)+" ■        ■ " NEXT RETURN
1290
1300 L$(1)=L$(1)+" ■      ■ "
1305 L$(2)=L$(2)+"■        ■"
1310 L$(3)=L$(3)+"■ ■    ■■ ■"
1315 L$(4)=L$(4)+"■■      ■ ■"
1320 L$(5)=L$(5)+"■       ■ ■"
1325 L$(6)=L$(6)+"■     ■ ■"
1330 L$(7)=L$(7)+"■   ■   ■"
1335 L$(8)=L$(8)+"    ■   ■"
1340 L$(9)=L$(9)+"■     ■■ ■"
1345 FORI=10TO11 L$(I)=L$(I)+"■       ■ ■" NEXTI
1350 L$(12)=L$(12)+"■■     ■ ■"
1355 L$(13)=L$(13)+"■ ■    ■ ■"
1360 L$(14)=L$(14)+"■■      ■"
1365 L$(15)=L$(15)+" ■■     ■ " RETURN
1390
1400 L$(1)=L$(1)+"      ■ ■"
1405 L$(2)=L$(2)+"     ■ ■ "
1410 L$(3)=L$(3)+"    ■ ■  "
1415 L$(4)=L$(4)+"   ■ ■   "
1420 L$(5)=L$(5)+"  ■ ■    "
1425 L$(6)=L$(6)+" ■ ■     "
1430 L$(7)=L$(7)+"■ ■      "
```

```
1435 L$(8)=L$(8)+"■ ■ ■"
1440 FORI=9TO10:L$(I)=L$(1)+"■ ■ ■ ■":NEXT
1445 FORI=11TO12:L$(I)=L$(I)+"■ ■":NEXT
1450 FORI=13TO15:L$(I)=L$(I)+"■ ■ ■ ■":NEXT:RETURN
1490 :
1500 L$(1)=L$(1)+"■ ■"
1502 L$(2)=L$(2)+"■ ■"
1505 FORI=3TO5:L$(I)=L$(I)+"■ ■ ■":NEXT
1510 L$(6)=L$(6)+"■ ■"
1515 L$(7)=L$(7)+"■ ■"
1520 L$(8)=L$(8)+"■ ■"
1525 L$(9)=L$(9)+"■ ■"
1530 FORI=10TO11:L$(I)=L$(I)+"■ ■ ■":NEXT
1535 L$(12)=L$(12)+"■ ■ ■"
1540 L$(13)=L$(13)+"■ ■ ■"
1545 L$(14)=L$(14)+"■ ■"
1550 L$(15)=L$(15)+"■ ■":RETURN
1590 :
1600 L$(1)=L$(1)+"■ ■"
1610 L$(2)=L$(2)+"■ ■"
1615 L$(3)=L$(3)+"■ ■"
1620 FORI=4TO7:L$(I)=L$(I)+"■ ■":NEXT
1625 L$(8)=L$(8)+"■ ■"
1630 L$(9)=L$(9)+"■ ■"
1635 L$(10)=L$(10)+"■ ■ ■"
1640 FORI=11TO12:L$(I)=L$(I)+"■ ■ ■":NEXT
1645 L$(13)=L$(13)+"■ ■"
1650 L$(14)=L$(14)+"■ ■"
1655 L$(15)=L$(15)+"■ ■":RETURN
1690 :
1700 L$(1)=L$(1)+"■ ■"
1705 L$(2)=L$(2)+"■ ■"
1710 L$(3)=L$(3)+"■ ■"
1715 L$(4)=L$(4)+"■ ■"
1720 L$(5)=L$(5)+"■ ■"
1725 L$(6)=L$(6)+"■ ■"
1730 L$(7)=L$(7)+"■ ■"
1735 L$(8)=L$(8)+"■ ■"
1740 L$(9)=L$(9)+"■ ■"
1745 L$(10)=L$(10)+"■ ■"
1750 L$(11)=L$(11)+"■ ■"
1755 L$(12)=L$(12)+"■ ■"
1760 L$(13)=L$(13)+"■ ■"
1765 L$(14)=L$(14)+"■ ■"
1770 L$(15)=L$(15)+"■ ■":RETURN
1790 :
1800 L$(1)=L$(1)+"■ ■"
1805 L$(2)=L$(2)+"■ ■"
1810 L$(3)=L$(3)+"■ ■ ■"
1815 L$(4)=L$(4)+"■ ■ ■"
1820 L$(5)=L$(5)+"■ ■ ■"
1822 L$(6)=L$(6)+"■ ■ ■"
1825 L$(7)=L$(7)+"■ ■"
1835 L$(8)=L$(8)+"■ ■"
1836 L$(9)=L$(9)+"■ ■ ■"
1840 FORI=10TO12:L$(I)=L$(I)+"■ ■ ■":NEXT
1845 L$(13)=L$(13)+"■ ■"
1850 L$(14)=L$(14)+"■ ■"
1855 L$(15)=L$(15)+"■ ■":RETURN
1890 :
1900 L$(1)=L$(1)+"■ ■"
1905 L$(2)=L$(2)+"■ ■"
1910 L$(3)=L$(3)+"■ ■ ■"
1912 L$(5)=L$(5)+"■ ■ ■"
1915 L$(4)=L$(4)+"■ ■ ■"
1916 L$(6)=L$(6)+"■ ■ ■"
1920 L$(7)=L$(7)+"■ ■"
1925 L$(8)=L$(8)+"■ ■"
1930 FORI=9TO12:L$(I)=L$(I)+"■ ■":NEXT
1935 L$(13)=L$(13)+"■ ■"
1940 L$(14)=L$(14)+"■ ■"
1945 L$(15)=L$(15)+"■ ■":RETURN
```

A program órája úgy működik, hogy az előbb említett szubrutin figyeli rajta a TI$ sztring perc, illetve másodperc-karaktereinek megváltozását. A másodpercek változása esetén a szubrutin hív egy másik szubrutint, amely az új pontos időt felírja a képernyő jobb felső részére. Ha a TI$ változó perc része változik meg, akkor a szubrutin hív egy olyan szubrutint, amely a hátralevő játékidőt írja ki. Ugyanez a TI$ változót ellenőrző szubrutin figyeli az F7 funkcióbillentyű megérintését is, amely lehetővé teszi az időmérés megszakítását és a visszaszámláló óra újbóli beállítását és újraindítását.

A program felépítése

5— 50 Képernyőtörlés, színbeállítás, feliratok. A 20-as sor átírásával a verseny neve vagy a csapatnevek jeleníthetők meg a képernyőn.

70—110 A játékidő lekérdezése, J és L változók beállítása. L a játékidő bal oldali, J pedig a jobb oldali karakterét tartalmazza.
A J változó tulajdonképpen a játékidő 10 perceket, L változó a perceket tartalmazza.

140—240 Az indítás időpontjának lekérdezése és az óra indítása.

280—370 A hátralevő játékidőt írja ki a szubrutin nagyméretű számjegyekkel.
A kiíratás az L$() sztringtömb feltöltésével kezdődik. Az 1000-es sortól kezdődő szubrutin tartalmazza a 0 karakter 15 sorra nagyított ábrázolását. Az 1100-tól az egyes, 1200-tól a kettes, . . . 1900-tól a 9-es számjegy 15 soros ábrázolása található.
Az 1000—1900-as szubrutinok úgy működnek, hogy L$(1)—L$(15)-ig hozzáadják a karakterjeleket az L$() sztring tartalmához.
Ha L$() tömb elemei üresek, akkor a 10 perceket jelölő első, egyébként a perceket jelentő, második karaktert helyezik el az L$() sztringtömbben.
Ez a módszer tette lehetővé, hogy a 0—9 számjegyek grafikus karaktereit csak egyszer tárolja a program, és a 280—370 szubrutin ezek segítségével összeállíthassa a mindenkori kijelzendő kétkarakteres játékidőt.

390—430 A másodpercek és percek változását érzékelő szubrutin. Ez a program órája.
A másodpercek változását T$ és TI$ összehasonlításával érzékeli. Változás esetén a pontos időt felírja a képernyő jobb felső részére. A percek változását TA$ és HA$ összehasonlításával érzékeli. Változás esetén a visszalévő játékidőt írja ki.

450—460 Ez a szubrutin írja ki a pontos időt a képernyő jobb felső részére. A kiíráskor az óra, perc, másodperc értékek közé kettőspontot tesz.

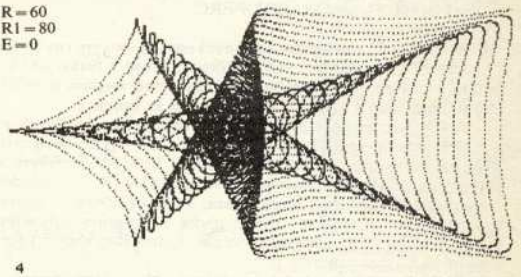
1000—1040 A 0 számjegy 15 sorra nagyított grafikus képe.

1100—11 . .1200—12 . . , 1300—13 . . , 1900—19 . . az 1-es 2-es . . . 9-es számjegyek grafikus képe.

SCHELL FERENC

COMMODORE 64

Rajzolás



4

```
10 PRINT"R"
20 INPUTR
30 PRINT"R1"
40 INPUTR1
50 PRINT"E"
60 INPUTE
90 C=R1-R
100 HIRES0,1
110 FORK=80 TO 300 STEP5
120 FORS=1 TO1500 STEP5
130 A=S/R1
160 K1=K+C*SIN(A)
170 K2=90-C*COS(A)
180 A2=S/R
190 G=A2+(π/2)-A
200 D=(π-G)
210 X=K1-COS(D)*(R-E)
215 IFX<1 THEN  260
218 IFX>355 THEN 260
220 Y=K2-SIN(D)*(R-E)
230 A$="5"
240 ROT0,1
250 DRAWA$,X,Y,1
260 NEXTS
265 E=E+5
270 NEXTK
```

A Simon's BASIC utasításait felhasználva, saját fantáziámból készítettem ezt az ábrát. Az alakzatot R, R1 és E megadásával lehet változtatni. Természetesen úgy kell megadni ezek értékét, hogy az ábra ráférjen a képernyőre. A rajzolást a 110-es lépésben levő STEP után álló szám növelésével lehet gyorsítani. Ezáltal azonban kevesebb pontból fog összeállni az ábra. A TO utáni szám nagyságát a megadott paraméterekből én számoltam ki.

SIPOS GYÖRGY

Programgyorsító fogások

A nyugatnémet COMMODORE újság 1984. októberi számában érdekes cikket közöl a BASIC programok futási ideje és azok külső megjelenési formája közti összefüggésekről. Eszerint néhány – a program lényegét nem érintő – fogást alkalmazva a program működése jelentősen felgyorsulhat.

A cikken felbuzdulva mintaprogramot készítettünk, amelyben kipróbáltuk a felsorolt fogásokat. Kísérletünk eredményéről számolunk be.

A kiinduló program a következő:

```
10 TI$="000000"
20 PRINT CHR$ (147)
30 Z=0
40 POKE 1024+Z,1:POKE 55296+Z,14
50 Z=Z+1
60 IF Z=600 THEN 1000
70 GOTO 40
1000 PRINT TAB (240); TAB (240);TAB(240)
     :PRINT TI/60 "MÁSODPERC"
```

Mint látjuk, a program alaptevékenysége a képernyő első 15 sorának A karakterrel való feltöltése, majd a futási idő kiiratása a képernyő 18. sorába. A példa természetesen szemléltető jellegű, nem a konkrét tevékenysége az érdekes.

A feltöltés az 1024-es képernyő-puffercímen kezdődik és a POKE utasítás segítségével 6 karakter (15 × 40) hosszan tart. Vele párhuzamosan az 55296-os címtől kezdődően folyik a karakterek színének világoskékre változtatása. A TI$ az időmérés kezdeti értékének beállítására, a 147 printelése a képernyő törlésére, az 1000-es sorban pedig a program működési idejének képernyőre íratására szolgál. A program ebben a formában 16,57 másodpercig tart.

Programgyorsító tevékenységünk első lépéseként a POKE utasításban használt konstansértékeket változóba helyezzük:

```
30 Z=0:KEP=1024:SZIN=55296
40 POKE KEP+Z,1:POKE SZIN+2,14
```

Újraindítva a programot, a futási idő 11,37 másodperc, ami az eredetinek 68,6 százaléka.

Ha az A betűt, a szín karakterkódját és a Z ciklusváltozó végső értékét is változóba helyezzük:

```
30 Z=0:KEP=1024:SZIN=55296:BETU=1:KOD=14
   :VEGE=600
```

és megváltoztatjuk a program 40. és 60. sorát:

```
40 POKE KEP+Z,BETU:POKE SZIN+Z,KOD
60 IF VEGE=Z THEN 1000
```

akkor a futási idő 9,4 másodperc, az eredetinek 56,7 százaléka.

Talán feltűnő, hogy a változók neve hosszabb két karakternél. Ezt a Commodore megengedi, bár csak az első két karaktert használja a változó azonosítására, viszont az ilyen nevek megnövelik a futási időt. Nézzük tehát, mi történik ha a változóneveket lerövidítjük 2 karakterre. Ezt a KEP, BETU, KOD, SZIN, VEGE változókkal tehetjük meg. A képernyőn ismét egy kisebb szám, a 9,12 másodperc jelenik meg, ami az eredetinek 55 százaléka.

A két karaktert egyre változtatva, tovább rövidül a futási idő. A változtatás után programunk így fest:

```
30 K=1024:S=55296:B=1:Ø=14:V=600
40 POKE K+Z,B:POKE S+Z,Ø
60 IF Z=V THEN1000
```

A szükséges idő 8,92 másodperc (53,8 százalék).
Most cseréljük ki a POKE-ot PRINT-re a 40-es sorban:

```
40 PRINT CHR$ (65);
```

és szüntessük meg a TAB kurzor pozicionáló utasításokat az 1000-es sorban:

```
1000 PRINT TI/60 "MÁSODPERC"
```

A POKE-nek PRINT-re cserélése után a program 6,35 másodpercig fut (38,3 százalék).

Az A karakter ASCII kódjának (decimális) értékét változóba téve ismét rövidül az idő, 4,82 másodperc (29,1 százalék):

```
30 K=1024:S=55296:B=1:Ø=14:V=600:A$=CHR$(65)
40 PRINT A$
```

Ha az A karaktert szövegkonstansként a PRINT utasításba helyezzük, ismét rövidebb lesz a futási idő: 4,7 másodperc. Ez az eredetinek 28,4 százaléka.

```
400 PRINT "A";
```

Rövidül a futási idő akkor is, ha a 60-as sorban a THEN helyett a GOTO utasítást használjuk. Időeredményünk ekkor 4,68 másodperc (28,2 százalék).

Gyorsító hatása van az IF utasításban a negáció használatának is. Ha például a 60-as sorban az „egyenlő" helyett a „nem egyenlő" feltételre kérdezünk rá, azon túlmenően, hogy a 70-es sor szükségtelenné válik, tovább javul az időeredmény: 4,63 másodperc, 29,9 százalék.

Szinte forradalminak mondható az a változás, amit az IF-fel szervezett ciklusnak FOR-ral való kicserélése okoz:

```
30 FOR Z=Ø TO 600
50 NEXT Z
```

Az utasítással egyidejűleg szükségtelenné válik a 60-as sor. Az időmérés eredménye megdöbbentő: 1,68 másodperc, ami az eredetinek csupán a 10,1 százaléka.

Az utasításpár tovább gyorsítható, ha az 50-es sorban a NEXT után elhagyjuk a Z ciklusváltozót. Ez az egyszerű művelet az eredményt 1,53 másodpercre, 9,2 százalékra javítja.

A gyorsításnak egy másik módja az utasítások tömörítése egy sorba:

```
10 TI$="000000":PRINTCHR$(174):FORZ=1TO600
   :PRINT "A";:NEXT:PRINTTI/60"MÁSODPERC"
```

A mérés eredménye 1,47 másodperc, 8,87 százalék.

Mint látjuk, ezekkel az egyszerű változtatásokkal 16,7 másodpercről 1,47 másodpercre gyorsítottuk a program futását.

Programozás közben gyakran előfordul, hogy egy részproblémát többféleképpen is meg tudunk oldani, tulajdonképpen érdektelennek látszik, hogy melyik megoldást választjuk. A felsorolt példák viszont éppen azt bizonyítják, hogy érdemes figyelni arra, hogy melyik a gyorsabb utasítás, hiszen a végrehajtási idők közötti eltérés igen jelentős is lehet.

RICHTER JÖRG—GOSZTOLA FERENC

A 65XX MIKROPROCESSZOR GÉPI ÉS ASSEMBLER SZINTŰ UTASÍTÁSKÉSZLETE

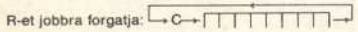
ÖSSZEÁLLÍTOTTA: ÉNEKES FERENC

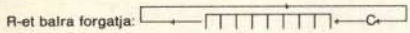
JELÖLÉSEK:

65XX REGISZTEREI:			SR BITJEI:
7 0 XXXXXXXX	A	akkumulátor	7. = N előjel bit
7 0 XXXXXXXX	Y	indexregiszter	6. = V belső átvitel bit
7 0 XXXXXXXX	X	indexregiszter	5. = – nem használt!
15 7 0 XX–HI–XXXX–LO–XX	PC	utasításszámláló	4. = B belső megszakítás bit
87 0 1XXXXXXX	SP	verem mutató	3. = D decimális számítási mód
7 0 NV–BDIZC	SR	állapotregiszter	2. = I megszakítás engedélyezés
7 0 ZZZZZZZZ	M	a memória egy bájtja	1. = Z nulla bit 0. = C túlcsordulás bit

N = 1 ha a művelet eredménye negatív.
V = 1 ha R6.→R7. = 1.
B = 1 a BRK utasítás kiszolgálása alatt.
D = 1 ha BCD aritmetikában számol a gép.
I = 1 ha a maszkolható megszakítások nincsenek engedélyezve.
Z = 1 ha a művelet eredménye nulla.
C = 1 ha R7.→R8. = 1.

R-et jobbra tolja: 0→ □□□□□□□□ →C
R-et balra tolja: C← □□□□□□□□ ←0





CÍMZÉSI MÓDOK

A 65XX processzor utasításai 1–3 bájt hosszúak. Az első bájt mindig meghatározza az utasáskódot és a címzési módot. A további bájtok a cím előállításához kellenek.

NÉV:	JEL:	MAGYARÁZAT:
1. IMPLICIT		Az operandust is a kód határozza meg. Pl.: PHP utasítás.
2. KÖZVETLEN	#ERTEK	A 2. bájt adat! Pl.: LDA $1B utasítás hatása A = 1B.
3. RELATÍV	(MOD.)	A 2. bájt 2-es komplemensben értelmezett relatív cím, mely PC aktuális (a következő utasítás első bájtjára mutató) értékéhez adódik hozzá. Pl.: BEQ $24 34-et viszi előre PC-t, míg BEQ $FF 126-ot hátra.
		–128(=(MOD.) (=+127, azaz max. 126-ot ugorhatunk hátra, illetve 129-et előre az aktuális utasítást követő utasítás első bájtjához viszonyítva.
4. 0 LAPOS	ZP	A 2. bájt a 0. lapon lévő címet jelenti. Pl.: LDA $2A hatására az akkumulátor a $002A cím tartalmával lesz egyenlő.
5. 0. LAPOS X-SZEL	ZP, X	A 2. bájt értékéhez hozzáadódik X tartalma. Pl.: ha végrehajtáskor X=$12, akkor az LDA $55,X utasítás A-ba a $0067 cím tartalmát tölti.
6. 0. LAPOS Y-NAL	ZP, Y	Azonos az előzővel, csak X helyett Y-t használja.
7. INDEXELT INDIREKT	(ZP, X)	Csak X használható! Az utasítás 2. bájtja hozzáadódik CARRY-vel az X regiszter tartalmához. Az eredmény egy 2 bájtos cím LO bájtjára mutat a 0. lapon. A cím HI bájtja a 0. lap következő bájtján van. Pl.: legyen X=$12, valamint ($26)=5A és ($27)=1B. Az LDA ($14,X) utasítás A-ba a $1B5A cím tartalmát tölti. ($14+$12=$26)
8. INDIREKT INDEXELT	(ZP), Y	Csak Y használható! Az utasítás 2. bájtja egy cím a 0. lapon, amelynek tartalma hozzáadódik az Y regiszter tartalmához. Az eredmény az aktuális cím LO bájtja. Az összeadáskor keletkezett CARRY a következő cím tartalmához adódik hozzá. Az eredmény az aktuális cím HI bájtja. Pl.: legyen Y=$12, valamint ($26)=5A és ($27)=1B. Az LDA ($26), Y utasítás A-ba a $1B6C cím tartalmát tölti. ($5A+$12=$6C)
9. ABSZOLÚT	ABS.	A 2. és 3. bájt egy 16 bites címet határoz meg. Pl.: az LDA $1234 utasítás A-ba a $1234 cím tartalmát tölti.
10. ABSZOLÚT X-SZEL	ABS., X	Az aktuális cím úgy áll elő, hogy az utasítás 2. és 3. bájtjához hozzáadjuk X tartalmát. Pl.: legyen X=$12, akkor az LDA $31AB utasítás a $31BD cím tartalmát tölti A-ba. ($AB+$12=$BD)
11. ABSZOLÚT Y-NAL	ABS., Y	Azonos az előzővel, csak X helyett Y-t használja.
12. ABSZOLÚT INDIREKT	(ABS.)	Csak JMP-nal használt! Pl.: JMP ($FF03) hatására PC-be töltődik a $FF03 cím tartalma és a program futása onnan folytatódik.

TOVÁBBI JELÖLÉSEK

+ : az ugrás végrehajtása esetén még egy ciklus kell.
• : amennyiben a címzés módosítása laphatárt lép át, még egy ciklus kell.
R : A illetve M lehet.
6 : M 6. bitje.
7 : M 7. bitje.
X : SR aktuális bitjének értéke a művelet eredményétől függ.
– : SR aktuális bitjének értékét nem befolyásolja a művelet.

A 65XX UTASÍTÁSKÉSZLETE NÉV SZERINT RENDEZVE

NÉV	CÍMZÉS	HEXA	DECI	NV-BDIZC	HO	CIK	FUNKCIO
ADC	(ZP,X)	61	097	XX----XX	2	6	A=A+M+C C=1 HA A7.->A8.=1
ADC	ZP	65	101	XX----XX	2	3	A=A+M+C C=1 HA A7.->A8.=1
ADC	#ERTEK	69	105	XX----XX	2	2	A=A+M+C C=1 HA A7.->A8.=1
ADC	ABS.	6D	109	XX----XX	3	4	A=A+M+C C=1 HA A7.->A8.=1
ADC	(ZP),Y	71	113	XX----XX	2	5*	A=A+M+C C=1 HA A7.->A8.=1
ADC	ZP,X	75	117	XX----XX	2	4	A=A+M+C C=1 HA A7.->A8.=1
ADC	ABS.,Y	79	121	XX----XX	3	4*	A=A+M+C C=1 HA A7.->A8.=1
ADC	ABS.,X	7D	125	XX----XX	3	4*	A=A+M+C C=1 HA A7.->A8.=1
AND	(ZP,X)	21	033	X-----X-	2	6	A=A AND M
AND	ZP	25	037	X-----X-	2	3	A=A AND M
AND	#ERTEK	29	041	X-----X-	2	2	A=A AND M
AND	ABS.	2D	045	X-----X-	3	4	A=A AND M
AND	(ZP),Y	31	049	X-----X-	2	5	A=A AND M
AND	ZP,X	35	053	X-----X-	2	4	A=A AND M
AND	ABS.,Y	39	057	X-----X-	3	4*	A=A AND M
AND	ABS.,X	3D	061	X-----X-	3	4*	A=A AND M
ASL	ZP	06	006	X-----XX	2	5	BALRA TOLAS 1 BITTEL
ASL	-	0A	010	X-----XX	1	2	BALRA TOLAS 1 BITTEL
ASL	ABS.	0E	014	X-----XX	3	6	BALRA TOLAS 1 BITTEL
ASL	ZP,X	16	022	X-----XX	2	6	BALRA TOLAS 1 BITTEL
ASL	ABS.,X	1E	030	X-----XX	3	7	BALRA TOLAS 1 BITTEL
BCC	<MOD.>	90	144	--------	2	2**	UGRAS HA C=0
BCS	<MOD.>	B0	176	--------	2	2**	UGRAS HA C=1
BEQ	<MOD.>	F0	240	--------	2	2**	UGRAS HA Z=1
BIT	ZP	24	036	76-----X-	2	3	Z=1 HA A AND M=0 N=M7. V=M6.
BIT	ABS.	2C	044	76-----X-	2	4	Z=1 HA A AND M=0 N=M7. V=M6.
BMI	<MOD.>	30	048	--------	2	2**	UGRAS HA N=1
BNE	<MOD.>	D0	208	--------	2	2**	UGRAS HA Z=0
BPL	<MOD.>	10	016	--------	2	2**	UGRAS HA N=0
BRK		00	000	---1--1X-	1	7	PC ES SR MENTESE UTAN STOP
BVC	<MOD.>	50	080	--------	2	2**	UGRAS HA V=0
BVS	<MOD.>	70	112	--------	2	2**	UGRAS HA V=1
CLC		18	024	-------0	1	2	C=0
CLD		D8	216	-----0--	1	2	D=0 BINARIS ARITMETIKA HASZNALAT
CLI		58	088	-----0--	1	2	I=0 MASZKOLHATO MEGSZAKITAS ENGED.
CLV		B8	184	-0------	1	2	V=0
CMP	(ZP,X)	C1	193	X-----XX	2	6	SR ALLAPOTA A-M FUGGVENYE
CMP	ZP	C5	197	X-----XX	2	3	SR ALLAPOTA A-M FUGGVENYE
CMP	#ERTEK	C9	201	X-----XX	2	2	SR ALLAPOTA A-M FUGGVENYE
CMP	ABS.	CD	205	X-----XX	3	4	SR ALLAPOTA A-M FUGGVENYE
CMP	(ZP),Y	D1	209	X-----XX	2	5*	SR ALLAPOTA A-M FUGGVENYE
CMP	ZP,X	D5	213	X-----XX	2	4	SR ALLAPOTA A-M FUGGVENYE
CMP	ABS.,Y	D9	217	X-----XX	3	4*	SR ALLAPOTA A-M FUGGVENYE
CMP	ABS.,X	DD	221	X-----XX	3	4*	SR ALLAPOTA A-M FUGGVENYE
CPX	#ERTEK	E0	224	X-----XX	2	2	SR ALLAPOTA X-M FUGGVENYE
CPX	ZP	E4	228	X-----XX	2	3	SR ALLAPOTA X-M FUGGVENYE
CPX	ABS.	EC	236	X-----XX	3	4	SR ALLAPOTA X-M FUGGVENYE
CPY	#ERTEK	C0	192	X-----XX	2	2	SR ALLAPOTA Y-M FUGGVENYE
CPY	ZP	C4	196	X-----XX	2	3	SR ALLAPOTA Y-M FUGGVENYE
CPY	ABS.	CC	204	X-----XX	3	4	SR ALLAPOTA Y-M FUGGVENYE
DEC	ZP	C6	198	X-----X-	2	5	M=M-1
DEC	ABS.	CE	206	X-----X-	3	6	M=M-1
DEC	ZP,X	D6	214	X-----X-	2	6	M=M-1
DEC	ABS.,X	DE	222	X-----X-	3	7	M=M-1
DEX		CA	202	X-----X-	1	2	X=X-1
DEY		88	136	X-----X-	1	2	Y=Y-1
EOR	(ZP,X)	41	065	X-----X-	2	6	A=A EOR M KIZARO VAGY
EOR	ZP	45	069	X-----X-	2	3	A=A EOR M KIZARO VAGY
EOR	#ERTEK	49	073	X-----X-	2	2	A=A EOR M KIZARO VAGY
EOR	ABS.	4D	077	X-----X-	3	4	A=A EOR M KIZARO VAGY
EOR	(ZP),Y	51	081	X-----X-	2	5*	A=A EOR M KIZARO VAGY
EOR	ZP,X	55	085	X-----X-	2	4	A=A EOR M KIZARO VAGY
EOR	ABS.,Y	59	089	X-----X-	3	4*	A=A EOR M KIZARO VAGY
EOR	ABS.,X	5D	093	X-----X-	3	4*	A=A EOR M KIZARO VAGY
INC	ZP	E6	230	X-----X-	2	5	M=M+1
INC	ABS.	EE	238	X-----X-	3	6	M=M+1
INC	ZP,X	F6	246	X-----X-	2	6	M=M+1
INC	ABS.,X	FE	254	X-----X-	3	7	M=M+1
INX		E8	232	X-----X-	1	2	X=X+1
INY		C8	200	X-----X-	1	2	Y=Y+1
JMP	ABS.	4C	076	--------	3	3	FELTETEL NELKULI UGRAS
JMP	(ABS.)	6C	108	--------	3	3	FELTETEL NELKULI UGRAS
JSR	ABS.	20	032	--------	3	6	UGRAS SZUBRUTINRA
LDA	(ZP,X)	A1	161	X-----X-	2	6	A=M
LDA	ZP	A5	165	X-----X-	2	3	A=M
LDA	#ERTEK	A9	169	X-----X-	2	2	A=M
LDA	ABS.	AD	173	X-----X-	3	4	A=M
LDA	(ZP),Y	B1	177	X-----X-	2	5*	A=M
LDA	ZP,X	B5	181	X-----X-	2	4	A=M
LDA	ABS.,Y	B9	185	X-----X-	3	4*	A=M
LDA	ABS.,X	BD	189	X-----X-	3	4*	A=M
LDX	#ERTEK	A2	162	X-----X-	2	2	X=M
LDX	ZP	A6	166	X-----X-	2	3	X=M
LDX	ABS.	AE	174	X-----X-	3	4	X=M
LDX	ZP,Y	B6	182	X-----X-	2	4	X=M
LDX	ABS.,Y	BE	190	X-----X-	3	4*	X=M
LDY	#ERTEK	A0	160	X-----X-	2	2	Y=M
LDY	ZP	A4	164	X-----X-	2	3	Y=M
LDY	ABS.	AC	172	X-----X-	3	4	Y=M
LDY	ZP,X	B4	180	X-----X-	2	4	Y=M
LDY	ABS.,X	BC	188	X-----X-	3	4*	Y=M
LSR	ZP	46	070	0-----XX	2	5	JOBBRA TOLAS 1 BITTEL
LSR	-	4A	074	0-----XX	1	2	JOBBRA TOLAS 1 BITTEL
LSR	ABS.	4E	078	0-----XX	3	6	JOBBRA TOLAS 1 BITTEL
LSR	ZP,X	56	086	0-----XX	2	6	JOBBRA TOLAS 1 BITTEL
LSR	ABS.,X	5E	094	0-----XX	3	7	JOBBRA TOLAS 1 BITTEL
NOP		EA	234	--------	1	2	URES UTASITAS CIMKET VISELHET
ORA	(ZP,X)	01	001	X-----X-	2	6	A=A OR M MEGENGEDO VAGY
ORA	ZP	05	005	X-----X-	2	3	A=A OR M MEGENGEDO VAGY
ORA	#ERTEK	09	009	X-----X-	2	2	A=A OR M MEGENGEDO VAGY
ORA	ABS.	0D	013	X-----X-	3	4	A=A OR M MEGENGEDO VAGY
ORA	(ZP),Y	11	017	X-----X-	2	5	A=A OR M MEGENGEDO VAGY
ORA	ZP,X	15	021	X-----X-	2	4	A=A OR M MEGENGEDO VAGY
ORA	ABS.,Y	19	025	X-----X-	3	4*	A=A OR M MEGENGEDO VAGY
ORA	ABS.,X	1D	029	X-----X-	3	4*	A=A OR M MEGENGEDO VAGY
PHA		48	072	--------	1	3	A-T VEREMBE TOLT
PHP		08	008	--------	2	3	SR-T VEREMBE TOLT
PLA		68	104	X-----X-	2	4	VEREMBOL A-BA TOLT
PLP		28	040	XX-XXXXX	1	4	VEREMBOL SR-BE TOLT
ROL	ZP	26	038	X-----XX	2	5	FORGATAS BALRA 1 BITTEL
ROL	-	2A	042	X-----XX	1	2	FORGATAS BALRA 1 BITTEL
ROL	ABS.	2E	046	X-----XX	3	6	FORGATAS BALRA 1 BITTEL
ROL	ZP,X	36	054	X-----XX	2	6	FORGATAS BALRA 1 BITTEL
ROL	ABS.,X	3E	062	X-----XX	3	7	FORGATAS BALRA 1 BITTEL
ROR	ZP	66	102	X-----XX	2	5	FORGATAS JOBBRA 1 BITTEL
ROR	-	6A	106	X-----XX	1	2	FORGATAS JOBBRA 1 BITTEL
ROR	ABS.	6E	110	X-----XX	3	6	FORGATAS JOBBRA 1 BITTEL
ROR	ZP,X	76	118	X-----XX	2	6	FORGATAS JOBBRA 1 BITTEL
ROR	ABS.,X	7E	126	X-----XX	3	7	FORGATAS JOBBRA 1 BITTEL
RTI		40	064	XX-XXXXX	1	6	VISSZATERES MEGSZAKITASBOL
RTS		60	096	--------	2	6	VISSZATERES SZUBRUTINBOL
SBC	(ZP,X)	E1	225	XX----XX	2	6	A=A-M-NEGALT C
SBC	ZP	E5	229	XX----XX	2	3	A=A-M-NEGALT C
SBC	#ERTEK	E9	233	XX----XX	2	2	A=A-M-NEGALT C
SBC	ABS.	ED	237	XX----XX	3	4	A=A-M-NEGALT C
SBC	(ZP),Y	F1	241	XX----XX	2	5*	A=A-M-NEGALT C
SBC	ZP,X	F5	245	XX----XX	3	4*	A=A-M-NEGALT C
SBC	ABS.,Y	F9	249	XX----XX	3	4*	A=A-M-NEGALT C
SBC	ABS.,X	FD	253	XX----XX	3	4*	A=A-M-NEGALT C
SEC		38	056	-------1	1	2	C=1
SED		F8	248	----1---	1	2	D=1 BCD ARITMETIKA HASZNALAT
SEI		78	120	-----1--	1	2	I=1 MASZKOLHATO MEGSZAKITAS TILTASA
STA	(ZP,X)	81	129	--------	2	6	M=A
STA	ZP	85	133	--------	2	3	M=A
STA	ABS.	8D	141	--------	3	4	M=A
STA	(ZP),Y	91	145	--------	2	6	M=A
STA	ZP,X	95	149	--------	2	4	M=A
STA	ABS.,Y	99	153	--------	3	5	M=A
STA	ABS.,X	9D	157	--------	3	5	M=A
STX	ZP	86	134	--------	2	3	M=X
STX	ABS.	8E	142	--------	3	4	M=X
STX	ZP,Y	96	150	--------	2	4	M=X
STY	ZP	84	132	--------	2	3	M=Y
STY	ABS.	8C	140	--------	3	4	M=Y
STY	ZP,X	94	148	--------	2	4	M=Y
TAX		AA	170	X-----X-	1	2	X=A
TAY		A8	168	X-----X-	1	2	Y=A
TSX		BA	186	X-----X-	1	2	X=SP
TXA		8A	138	X-----X-	1	2	A=X
TXS		9A	154	--------	1	2	SP=X
TYA		98	152	X-----X-	1	2	A=Y

HEXA	DECI	NEV	CIMZES	NV-BDIZC	HO	CIK	FUNKCIO
00	000	BRK		---1--X-	1	7	PC ES SR MENTESE UTAN STOP
01	001	ORA	(ZP,X)	X-----X-	2	6	A=A OR M MEGENGEDO VAGY
05	005	ORA	ZP	X-----X-	2	3	A=A OR M MEGENGEDO VAGY
06	006	ASL	ZP	X-----XX	2	5	BALRA TOLAS 1 BITTEL
08	008	PHP		--------	2	3	SR-T VEREMBE TOLT
09	009	ORA	#ERTEK	X-----X-	2	2	A=A OR M MEGENGEDO VAGY
0A	010	ASL		X-----XX	1	2	BALRA TOLAS 1 BITTEL
0D	013	ORA	ABS.	X-----X-	3	4	A=A OR M MEGENGEDO VAGY
0E	014	ASL	ABS.	X-----XX	3	6	BALRA TOLAS 1 BITTEL
10	016	BPL	<MOD.>	--------	2	2**	UGRAS HA N=0
11	017	ORA	(ZP),Y	X-----X-	2	5	A=A OR M MEGENGEDO VAGY
15	021	ORA	ZP,X	X-----X-	2	4	A=A OR M MEGENGEDO VAGY
16	022	ASL	ZP,X	X-----XX	2	6	BALRA TOLAS 1 BITTEL
18	024	CLC		-------0	1	2	C=0
19	025	ORA	ABS.,Y	X-----X-	3	4*	A=A OR M MEGENGEDO VAGY
1D	029	ORA	ABS.,X	X-----X-	3	4*	A=A OR M MEGENGEDO VAGY
1E	030	ASL	ABS.,X	X-----XX	3	7	BALRA TOLAS 1 BITTEL
20	032	JSR	ABS.	--------	3	6	UGRAS SZUBRUTINRA
21	033	AND	(ZP,X)	X-----X-	2	6	A=A AND M
24	036	BIT	ZP	76-----X	2	3	Z=1 HA A AND M=0 N=M7. V=M6.
25	037	AND	ZP	X-----X-	2	3	A=A AND M
26	038	ROL	ZP	X-----XX	2	5	FORGATAS BALRA 1 BITTEL
28	040	PLP		XX-XXXXX	1	4	VEREMBOL SR-BE TOLT
29	041	AND	#ERTEK	X-----X-	2	2	A=A AND M
2A	042	ROL		X-----XX	1	2	FORGATAS BALRA 1 BITTEL
2C	044	BIT	ABS.	76-----X	3	4	Z=1 HA A AND M=0 N=M7. V=M6.
2D	045	AND	ABS.	X-----X-	3	4	A=A AND M
2E	046	ROL	ABS.	X-----XX	3	6	FORGATAS BALRA 1 BITTEL
30	048	BMI	<MOD.>	--------	2	2**	UGRAS HA N=1
31	049	AND	(ZP),Y	X-----X-	2	5	A=A AND M
35	053	AND	ZP,X	X-----X-	2	4	A=A AND M
36	054	ROL	ZP,X	X-----XX	2	6	FORGATAS BALRA 1 BITTEL
38	056	SEC		-------1	1	2	C=1
39	057	AND	ABS.,Y	X-----X-	3	4*	A=A AND M
3D	061	AND	ABS.,X	X-----X-	3	4*	A=A AND M
3E	062	ROL	ABS.,X	X-----XX	3	7	FORGATAS BALRA 1 BITTEL
40	064	RTI		XX-XXXXX	1	6	VISSZTERES MEGSZAKITASBOL
41	065	EOR	(ZP,X)	X-----X-	2	6	A=A EOR M KIZARO VAGY
45	069	EOR	ZP	X-----X-	2	3	A=A EOR M KIZARO VAGY
46	070	LSR	ZP	0-----XX	2	5	JOBBRA TOLAS 1 BITTEL
48	072	PHA		--------	1	3	A-T VEREMBE TOLT
49	073	EOR	#ERTEK	X-----X-	2	2	A=A EOR M KIZARO VAGY
4A	074	LSR		0-----XX	1	2	JOBBRA TOLAS 1 BITTEL
4C	076	JMP	ABS.	--------	3	3	FELTETEL NELKULI UGRAS
4D	077	EOR	ABS.	X-----X-	3	4	A=A EOR M KIZARO VAGY
4E	078	LSR	ABS.	0-----XX	3	6	JOBBRA TOLAS 1 BITTEL
50	080	BVC	<MOD.>	--------	2	2**	UGRAS HA V=0
51	081	EOR	(ZP),Y	X-----X-	2	5*	A=A EOR M KIZARO VAGY
55	085	EOR	ZP,X	X-----X-	2	4	A=A EOR M KIZARO VAGY
56	086	LSR	ZP,X	0-----XX	2	6	JOBBRA TOLAS 1 BITTEL
58	088	CLI		-----0--	1	2	I=0 MASZKOLHATO MEGSZAKITAS ENGED.
59	089	EOR	ABS.,Y	X-----X-	3	4*	A=A EOR M KIZARO VAGY
5D	093	EOR	ABS.,X	X-----X-	3	4*	A=A EOR M KIZARO VAGY
5E	094	LSR	ABS.,X	0-----XX	3	7	JOBBRA TOLAS 1 BITTEL
60	096	RTS		--------	2	6	VISSZTERES SZUBRUTINBOL
61	097	ADC	(ZP,X)	XX----XX	2	6	A=A+M+C C=1 HA A7.->A8.=1
65	101	ADC	ZP	XX----XX	2	3	A=A+M+C C=1 HA A7.->A8.=1
66	102	ROR	ZP	X-----XX	2	5	FORGATAS JOBBRA 1 BITTEL
68	104	PLA		X-----X-	2	4	VEREMBOL A-BA TOLT
69	105	ADC	#ERTEK	XX----XX	2	2	A=A+M+C C=1 HA A7.->A8.=1
6A	106	ROR		X-----XX	1	2	FORGATAS JOBBRA 1 BITTEL
6C	108	JMP	(ABS.)	--------	3	5	FELTETEL NELKULI UGRAS
6D	109	ADC	ABS.	XX----XX	3	4	A=A+M+C C=1 HA A7.->A8.=1
6E	110	ROR	ABS.	X-----XX	3	6	FORGATAS JOBBRA 1 BITTEL
70	112	BVS	<MOD.>	--------	2	2**	UGRAS HA V=1
71	113	ADC	(ZP),Y	XX----XX	2	5*	A=A+M+C C=1 HA A7.->A8.=1
75	117	ADC	ZP,X	XX----XX	2	4	A=A+M+C C=1 HA A7.->A8.=1
76	118	ROR	ZP,X	X-----XX	2	6	FORGATAS JOBBRA 1 BITTEL
78	120	SEI		-----1--	1	2	I=1 MASZKOLHATO MEGSZAKITAS TILTASA
79	121	ADC	ABS.,Y	XX----XX	3	4*	A=A+M+C C=1 HA A7.->A8.=1
7D	125	ADC	ABS.,X	XX----XX	3	4*	A=A+M+C C=1 HA A7.->A8.=1
7E	126	ROR	ABS.,X	X-----XX	3	7	FORGATAS JOBBRA 1 BITTEL
81	129	STA	(ZP,X)	--------	2	6	M=A
84	132	STY	ZP	--------	2	3	M=Y
85	133	STA	ZP	--------	2	3	M=A

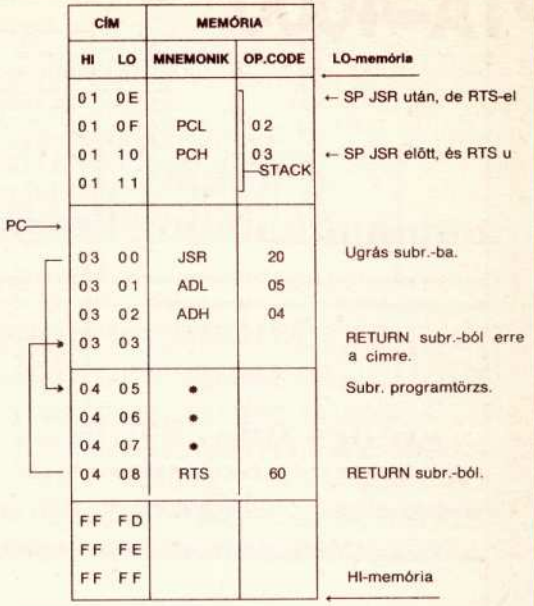
HEXA	DECI	NEV	CIMZES	NV-BDIZC	HO	CIK	FUNKCIO
86	134	STX	ZP	--------	2	3	M=X
88	136	DEY		X-----X-	1	2	Y=Y-1
8A	138	TXA		X-----X-	1	2	A=X
8C	140	STY	ABS.	--------	3	4	M=Y
8D	141	STA	ABS.	--------	3	4	M=A
8E	142	STX	ABS.	--------	3	4	M=X
90	144	BCC	<MUD.>	--------	2	2**	UGRAS HA C=0
91	145	STA	(ZP),Y	--------	2	6	M=A
94	148	STY	ZP,X	--------	2	4	M=Y
95	149	STA	ZP,X	--------	2	4	M=A
96	150	STX	ZP,Y	--------	2	4	M=X
98	152	TYA		X-----X-	1	2	A=Y
99	153	STA	ABS.,Y	--------	3	5	M=A
9A	154	TXS		--------	1	2	SP=X
9D	157	STA	ABS.,X	--------	3	5	M=A
A0	160	LDY	#ERTEK	X-----X-	2	2	Y=M
A1	161	LDA	(ZP,X)	X-----X-	2	6	A=M
A2	162	LDX	#ERTEK	X-----X-	2	2	X=M
A4	164	LDY	ZP	X-----X-	2	3	Y=M
A5	165	LDA	ZP	X-----X-	2	3	A=M
A6	166	LDX	ZP	X-----X-	2	3	X=M
A8	168	TAY		X-----X-	1	2	Y=A
A9	169	LDA	#ERTEK	X-----X-	2	2	A=M
AA	170	TAX		X-----X-	1	2	X=A
AC	172	LDY	ABS.	X-----X-	3	4	Y=M
AD	173	LDA	ABS.	X-----X-	3	4	A=M
AE	174	LDX	ABS.	X-----X-	3	4	X=M
B0	176	BCS	<MOD.>	--------	2	2**	UGRAS HA C=1
B1	177	LDA	(ZP),Y	X-----X-	2	5*	A=M
B4	180	LDY	ZP,X	X-----X-	2	4	Y=M
B5	181	LDA	ZP,X	X-----X-	2	4	A=M
B6	182	LDX	ZP,Y	X-----X-	2	4	X=M
B8	184	CLV		-0------	1	2	V=0
B9	185	LDA	ABS.,Y	X-----X-	3	4*	A=M
BA	186	TSX		X-----X-	1	2	X=SP
BC	188	LDY	ABS.,X	X-----X-	3	4*	Y=M
BD	189	LDA	ABS.,X	X-----X-	3	4*	A=M
BE	190	LDX	ABS.,Y	X-----X-	3	4*	X=M
C0	192	CPY	#ERTEK	X-----XX	2	2	SR ALLAPOTA Y-M FUGGVENYE
C1	193	CMP	(ZP,X)	X-----XX	2	6	SR ALLAPOTA A-M FUGGVENYE
C4	196	CPY	ZP	X-----XX	2	3	SR ALLAPOTA Y-M FUGGVENYE
C5	197	CMP	ZP	X-----XX	2	3	SR ALLAPOTA A-M FUGGVENYE
C6	198	DEC	ZP	X-----X-	2	5	M=M-1
C8	200	INY		X-----X-	1	2	Y=Y+1
C9	201	CMP	#ERTEK	X-----XX	2	2	SR ALLAPOTA A-M FUGGVENYE
CA	202	DEX		X-----X-	1	2	X=X-1
CC	204	CPY	ABS.	X-----XX	3	4	SR ALLAPOTA Y-M FUGGVENYE
CD	205	CMP	ABS.	X-----XX	3	4	SR ALLAPOTA A-M FUGGVENYE
CE	206	DEC	ABS.	X-----X-	3	6	M=M-1
D0	208	BNE	<MUD.>	--------	2	2**	UGRAS HA Z=0
D1	209	CMP	(ZP),Y	X-----XX	2	5*	SR ALLAPOTA A-M FUGGVENYE
D5	213	CMP	ZP,X	X-----XX	2	4	SR ALLAPOTA A-M FUGGVENYE
D6	214	DEC	ZP,X	X-----X-	2	6	M=M-1
D8	216	CLD		-----0--	1	2	D=0 BINARIS ARITMETIKA HASZNALHAT
D9	217	CMP	ABS.,Y	X-----XX	3	4*	SR ALLAPOTA A-M FUGGVENYE
DD	221	CMP	ABS.,X	X-----XX	3	4*	SR ALLAPOTA A-M FUGGVENYE
DE	222	DEC	ABS.,X	X-----X-	3	7	M=M-1
E0	224	CPX	#ERTEK	X-----XX	2	2	SR ALLAPOTA X-M FUGGVENYE
E1	225	SBC	(ZP,X)	XX----XX	2	6	A=A-M-NEGALT C
E4	228	CPX	ZP	X-----XX	2	3	SR ALLAPOTA X-M FUGGVENYE
E5	229	SBC	ZP	XX----XX	2	3	A=A-M-NEGALT C
E6	230	INC	ZP	X-----X-	2	5	M=M+1
E8	232	INX		X-----X-	1	2	X=X+1
E9	233	SBC	#ERTEK	XX----XX	2	2	A=A-M-NEGALT C
EA	234	NOP		--------	1	2	URES UTASITAS CIMKET VISELHET
EC	236	CPX	ABS.	X-----XX	3	4	SR ALLAPOTA X-M FUGGVENYE
ED	237	SBC	ABS.	XX----XX	3	4	A=A-M-NEGALT C
EE	238	INC	ABS.	X-----X-	3	6	M=M+1
F0	240	BEQ	<MOD.>	--------	2	2**	UGRAS HA Z=1
F1	241	SBC	(ZP),Y	XX----XX	2	5	A=A-M-NEGALT C
F5	245	SBC	ZP,X	XX----XX	3	4*	A=A-M-NEGALT C
F6	246	INC	ZP,X	X-----X-	2	6	M=M+1
F8	248	SED		-----1--	1	2	D=1 BCD ARITMETIKA HASZNALHAT
F9	249	SBC	ABS.,Y	XX----XX	3	4*	A=A-M-NEGALT C
FD	253	SBC	ABS.,X	XX----XX	3	4*	A=A-M-NEGALT C
FE	254	INC	ABS.,X	X-----X-	3	7	M=M+1

A 65XX MIKROPROCESSZOR GÉPI ÉS ASSEMBLER SZINTŰ UTASÍTÁSKÉSZLETE

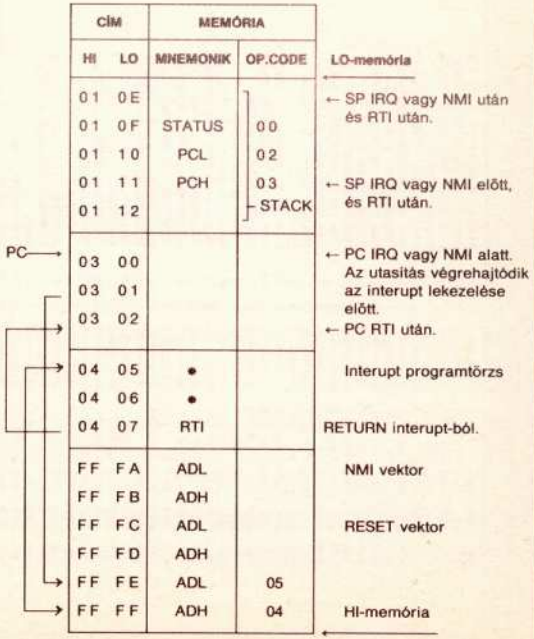
RÖVID ÖSSZEFOGLALÓ

NÉV	NV-BDIZC	FUNKCIÓ
ADC	XX——XX	A=A+M+C C=1 ha A7.→)A8.=1
AND	X——X–	A=A AND M
ASL	S——XX	Balra tolás 1 bittel
BCC	————	Ugrás ha C=0
BCS	————	Ugrás ha C=1
BEQ	————	Ugrás ha Z=1
BIT	76——X–	Z=1 ha A AND M=0 N=M7. V=M6.
BMI	————	Ugrás ha N=1
BNE	————	Ugrás ha Z=0
BPL	————	Ugrás ha N=0
BRK	——1—X–	PC és SR mentése után STOP
BVC	————	Ugrás ha V=0
BVS	————	Ugrás ha V=1
CLC	———0	C=0
CLD	——0——	D=0 bináris aritmetika használat
CLI	——0—	I=0 maszkolható megszakítás enged.
CLV	–0——	V=0
CMP	X——XX	SR állapota A–M függvénye
CPX	X——XX	SR állapota X–M függvénye
CPY	X——XX	SR állapota Y–M függvénye
DEC	X——X–	M=M–1
DEX	X——X–	X=X–1
DEY	X——X–	Y=Y–1
EOR	X——X–	A=A EOR M kizáró vagy
INC	X——X–	M=M+1
INX	X——X–	X=X+1
INY	X——X–	Y=Y+1
JMP	————	Feltétel nélküli ugrás
JSR	————	Ugrás szubrutinra
LDA	X——X–	A=M
LDX	X——X–	X=M
LDY	X——X–	Y=M
LSR	0——XX	Jobbra tolás 1 bittel
NOP	————	Üres utasítás címkét viselhet
ORA	X——X–	A=A OR M megengedő vagy
PHA	————	A-t verembe tölt
PHP	————	SR-t verembe tölt
PLA	X——X–	Veremből A-ba tölt
PLP	XX–XXXXX	Veremből SR-be tölt
ROL	X——XX	Forgatás balra 1 bittel
ROR	X——XX	Forgatás jobbra 1 bittel
RTI	XX–XXXXX	Visszatérés megszakításból
RTS	————	Visszatérés szubrutinból
SBC	XX——XX	A←A–M– negált C
SEC	———1	C=1
SED	——1——	D=1 BCD aritmetika használat
SEI	——1—	I=1 maszkolható megszakítás tiltása
STA	————	M=A
STX	————	M=X
STY	————	M=Y
TAX	X——X–	X=A
TAY	X——X–	Y=A
TSX	X——X–	X=SP
TXA	X——X–	A=X
TXS	————	SP=X
TYA	X——X–	A=Y

JSR ÉS RTS UTASÍTÁSOK VÉGREHAJTÁSA



RTI ÉS BRK UTASÍTÁS VÉGREHAJTÁSA



PTA-4000

A Híradástechnika Szövetkezet szerint, az általa Sharp-licenc alapján gyártott PTA—4000 típusú BASIC kalkulátorból már 1200 darabot használnak országszerte. Körülbelül ugyanennyi lehet az eredeti Sharp PC—1500 használók száma. Eddig egyik szaklapunk sem foglalkozott ezzel a géppel, bár engem is megkerestek felhasználók különböző kérdésekkel, például azzal is, hogy hol működik ilyen klub. Nem tudok ilyenről, de a HCC mindig kész volt bármilyen szekció létrehozására, ha az iránt kellő érdeklődés mutatkozott. Várom tehát a belépni szándékozó olvasók címmel, válaszborítékkal ellátott leveleit, hogy az alakuló ülés helyéről és időpontjáról értesítést küldhessek.

Lapunk a µKlub rovatban folyamatosan jelentet meg ezután PTA—4000-rel kapcsolatos anyagokat, amihez cikkeket várunk. Emellett igyekszünk a legnagyobb érdeklődést kiváltó témákról saját magunk írni.

Legtöbben a rendszer felépítését és a szoftverrendszert ismertető cikkeket hiányolják. Ilyeneket egy NSZK-beli klubban beszerzett anyag alapján fogok írni, az alábbi címekkel:

Az LH 5801 mikroprocesszor hardver
Az LH 5801 utasításkészlete
A hardver egyéb részei
BASIC kulcsszavak
A BASIC belső felépítése I—II.
Gépi kódú példák
ROM leírás I—III.

Az LH 5801 mikroprocesszor hardver

Első cikkünkből itt most a mikroprocesszor rövid áttekintő leírását adjuk. Az LH 5801 egy 8 bites CMOS mikroprocesszor, egybeépítve egy LCD (folyadékkristály kijelző) vezérlővel, egy tárolóműködtető órával és egy általános célú időzítővel.

Jellemzői:

128 k címterület (!)
6, egyenként 2 bájtos regiszter
9 bites időzítő
háromféle megszakításfípus
82 utasítás
WAIT-funkció
kimenő órajel
memóriaátmásolási lehetőség
közvetlen LCD-vezérlés
közvetlen kvarcbemenet
28 közvetlen, vektorszervezésű ugróutasítás

Az *1. ábrán* a mikroprocesszor blokkvázlata látható. A számokkal jelölt belső blokkok jelentése: 1 — címpuffer, 2 — oszcillátor, 3 — felező, 4 — órajelszabályzó, 5 — megszakításszabályzó, 6 — kijelző adatszámláló, 7 — regiszterek, 8 — 11 bites leosztó, 9 — kijelző adatkiadó és tápegységszabályzó, 10 — ALU, 11 — ütemadó-szabályzó, 12 — puffer, 13 — ütemadó-regiszter, 14 — utasításdekódoló és CPU-szabályzó, 15 — adatsínszabályzó, 16 — billenőkörök.

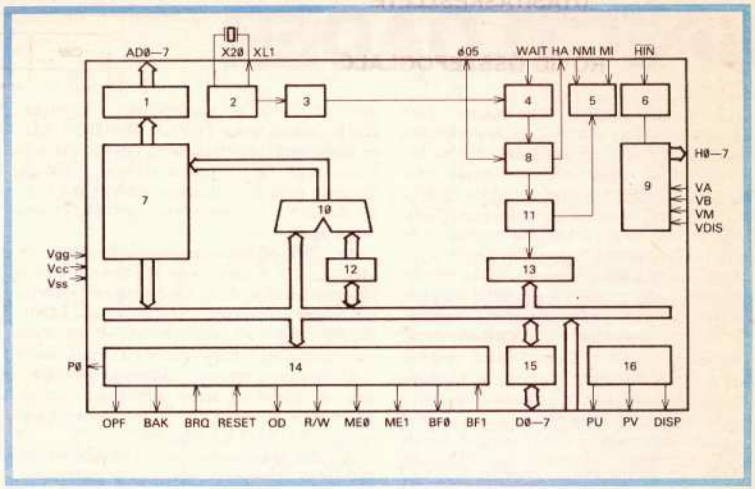
A *2. ábra* az egyes belső regisztereket mutatja. Itt P — a 16 bites programszámláló, S — a 16 bites veremmutató, X, Y, U — a 16 bites adatmutató, ill. általános célú regiszterek, A — a 8 bites akkumulátor, TM — a 9 bites ütemadó, PU, PV — 1 bites általános célú billenőkörök, DISP — 1 bites kijelző ki/be kapcsoló, T — állapotregiszter, C — átviteli/áthozati bit, IE — megszakításengedélyező bit, Z — zero bit, V — túlcsordulás bit, H — félátvileti/áthozati bit.

A 16 bites regiszterek két 8 biteskent is használhatók.

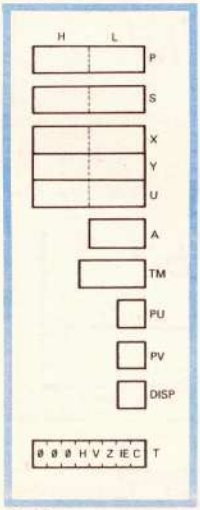
A *3. ábra* a mikroprocesszor ún. lábkiosztása. Az egyes jelek: XLØ,XL1 — a 2,6 MHz frekvenciájú kvarc jelei, ADØ-15 — címvezetékek, DØ-7 — adatvezetékek, MEØ-1 — memórialap-kiválasztó, ØOS — külső szinkronozó jel, R/W — írás/olvasásvezérlő, OD — adatkivitelt letiltó jel, RESET — újraindító, NMI — feltétel nélküli megszakító, MI — feltételes megszakító, BRQ — sínhasználat-engedélyezést kérő, BAK — az előző jel válaszjele, OPF — műveleti utasítást végrehajtó, INØ-7 — bemenő adatok, PØ — külső engedélyező jel, PU, PV — billenőkör-kimenetek, WAIT — leállítja a mikroprocesszort, HØ-7 — kijelzővezérlő adatok, VA,VB,VM,VDIS — tápvezetékek a kijelzőtől, HIN — számlálóbemenet, HA — a kijelző órajele, DISP — kijelző ki/bekapcsoló, BFO—BFI — billenőkör ki/bemenetek.

A jelek, vezetékek, részegységek áttekintő ismertetésére, mélyebb vizsgálatára itt nincs lehetőség, de amennyiben kellő számú érdeklődő jelentkezik, a HCC vállalja ilyen tanfolyam megrendezését.

DR. SIMONYI ENDRE



1. ábra



2. ábra

3. ábra

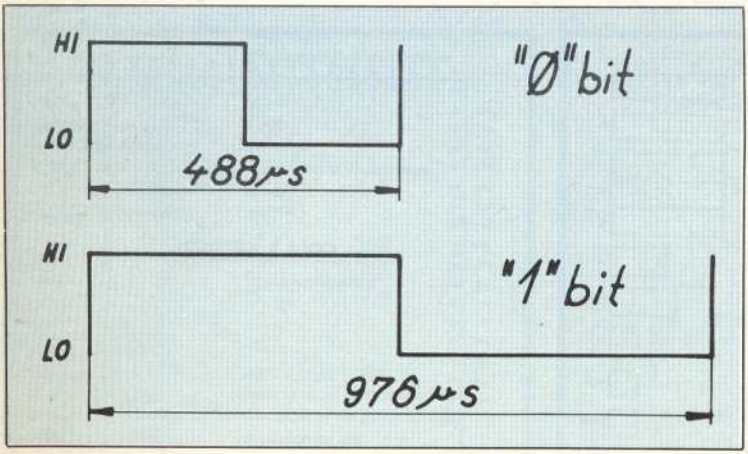
SPECTLOADER

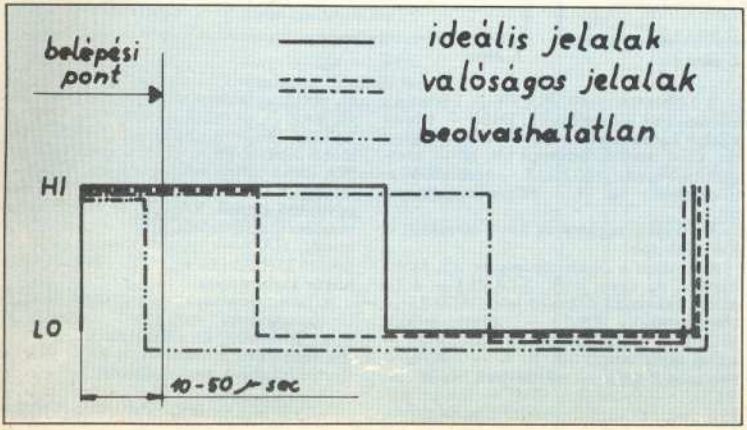
A Magyarországon forgalmazott Z80 mikroprocesszorra épülő személyi számítógépek közötti adat- és programcsere az inkompatibilitás miatt nagyon korlátozott. Az egyező processzortípus azonban kínálja azt a lehetőséget, hogy az egyik géptípusra készített, kisebb gépi kódú rutinokat más gépre átdolgozzuk, azon futtassuk. Az adatcsere egyik lehetősége az, ha az adatokat fogadó gép az idegen gép által magnószalagra eltárolt információkat megérti. A felvételek készítését a SAVER-ek, azok beolvasását a LOADER-ek végzik. Sajnos ezek a hasznos rutinok is gépenként különbözőek, így például a Spectrum nem „érti" a Homelab gépek felvételét.

Azok a felhasználók, akik rutinok átdolgozását kívánják elvégezni, két lehetőség között választhatnak. Vagy időt és fáradságot nem kímélve „bepötyögik" a klaviatúrán keresztül, vagy SAVER-LOADER írásra kényszerülnek. Ez utóbbi feladathoz szeretnék segítséget nyújtani azoknak, akik eddig még nem foglalkoztak a SAVER-LOADER-ekkel, de hasznosan tudnák azokat alkalmazni.

Az alábbiakban részletesen közölt rutin HOMELAB 2 (Aircomp 16) típusú számítógépre készült, és a ZX-Spectrum felvételét képes elolvasni. Eredetileg a LOPICOPY nevű S—L rutinom LOADER része volt, azonban a teljes rutin alkalmas lenne a Spectrum programok „tömeggyártására", ami a szoftvergyártók érdekeit sértené. Ezért csak a bonyolultabb és jobban használható LOADER-t közlöm.

A rutin ismertetése előtt röviden tekintsük át a Spectrum jeltárolási módszerét.





A Spectrum által készített felvétel két jól elkülöníthető részből áll.

A fejléc öt másodperc hosszúságú headerrel kezdődik, ami a szinkronizálást teszi lehetővé és 1238 µs periódusidejű jelsorozatból áll. A header végét egy Ø bit jelzi, ahonnan a jelfolyam bájtonként (8 bitenként) értelmezendő. Az első bájt a felvétel azonosítására szolgál. Ha az ØØH, akkor a LOADER fejlécet olvas, ha FFH, akkor adattömböt. A fejléc második bájtja meghatározza az utána következő felvétel típusát:

ØØH BASIC program
Ø1H numerikus blokk
Ø2H sztringblokk
Ø3H adatblokk

Ezután a következő tíz bájtba a kimentéskor megadott és maximálisan tíz karakterből álló név kerül. A 13. és 14. bájt a beolvasandó adatok számát tartalmazza, ahol a 13. az alsó, a 14. a felső bájt. A 15. és 16. bájt BASIC-nél az autostart kezdőcíme, egyébként a betöltési tárcím. A 17. és 18. bájt a BASIC tárterület hosszát adja. A 19. az úgynevezett CHECK, ellenőrző bájt, ami a fejléc összes adatának XOR függvénnyel képzett összege. Ha például a CHECK bájt nem egyezik az előző adatok XOR összegével, akkor a LOADER ERROR üzenetet küld, így elkerüli a hibás beolvasást.

Az adattömb két másodperc hosszúságú headerrel kezdődik, aminek jelalakja megegyezik a fejlécével. A Ø bit után az FFH azonosító következik, majd folyamatosan a fejléc által meghatározott számú adat és a végén az ellenőrző bájt.

A magnókimenetre kerülő elemi jelek szint—idő diagramját az *1. ábra* szemlélteti. Ezeket az elemi jeleket kapcsolja összefüggő bitfolyammá a SAVER.

A szalagra kerülő jeleket beolvasáskor a magnó többnyire eltorzítva adja vissza *(2. ábra)*, de a jel frekvenciája még a legigénytelenebb magnó esetén is 10% ingadozáson belül marad. Célszerű tehát, ha a LOADER egy teljes periódus hosszát vizsgálja. Egy elemi jel beolvasását a 2. ábra szemlélteti. A „belépési pont" az előző bit beolvasása utáni visszatérés helye. Ha a beolvasandó jel egyik éle — a torzítás miatt — a belül marad. Célszerű tehát, ha a LOADER egy teljes periódus hosszát vizsgálvetkezik. A jelek megfelelő arányainak beállítására alkalmas a µM 1985/5. számában közölt MINISZKOP program.

A rutin a HOMELAB 2 gép rendszerváltozói után 4Ø9ØH címre kerül, viszont a BASIC programok 4ØAØH-tól kezdődnek. Ebből az a kellemetlen helyzet alakul ki, hogy a LOADER „belelóg" a BASIC RAM területére. Ha tehát a rutin begépelése után belépünk BASIC-be, akkor a BASIC újra inicializálódik, tönkretéve a beírt rutint! Így az olyan HOMELAB 2 gépen, ami bekapcsoláskor BASIC-ben ébred, a RESET gomb sem használható. Ezzel a tárelrendezéssel viszont a lehető leghosszabb felhasz-

nálható RAM területet nyerjük. A LOÁDER indítása 4090 címről történik, monitor utasítással. Magnóra kivitt LOADER beolvasását is csak monitor utasításssal tudjuk elvégezni. A gép saját beolvasó rutinja a 0405H címen kezdődik, innen indítható szintén monitor utasítással.

A SPECTLOADER két gomb megnyomását figyeli. Az egyik az L gomb, ami a beolvasást indítja el. A bevett fejléc után a képszerkesztés visszatér, és kiírja a program nevét. Ha az adattömb mérete nagyobb a szabad tárterületnél, akkor OUT OF MEMORY üzenet is megjelenik a képernyőn. Ekkor még csak a fejléc beolvasása történt meg. Az L gomb újbóli megnyomása indítja tovább a LOADER-t. Ha a fejléc beolvasása után ERROR üzenet keletkezik, vagy nem kívánjuk beolvasni a kiírt című programot, akkor bármely más gombbal ezt jelezhetjük a gépnek, ami a rutin elejére áll. Ha a rutin valamilyen oknál fogva nem tér vissza, akkor az R gomb megnyomásával alaphelyzetbe állíthatjuk.

A részletes Assembler lista előtt azonban meg kell jegyeznem, hogy a HOMELAB 2 fordítórutinjának szintaktikája bizonyos mértékig eltér a Z80 ASSEMBLER-től. A különbségek nem túl nagyok, így az eltérések magyarázatára külön nem térek ki.

```
c1      MV      SP,$4090
        LD      A,$0D
        RST5
        RST3
        CP      $4C
        JPNZ    c1
        MV      ($3E3E),A       képszerkesztés
                                lekapcsolása
c2      CALL    "SINC"          szinkron rutin hívá-
                                sa
        AND     A
        JRNZ    c2              ha az azonosító
                                nem
                                00H, akkor
                                tovább
                                keresi a fejlécet
        MV      HL,"KEZD"
        MV      BC,$1200
c3      CALL    "BYTIN"
        LD      (HL),A
        XOR     C               fejléc+CHECK
                                bájt
        LD      C,A             beolvasása és az el-
                                len-
        INC     HL              őrző bájt képzése
        DJNZ    c3
        MV      ($3F3F),A       képszerkesztés vissz-
                                aállítása hiba esetén
                                ERROR üzenet hí-
                                vása
        CA-     "LERROR"
        LLNZ
        PUSH    HL
        MV      HL,"FREE"       szabad RAM terület
                                mérete a HL-be
        MV      DE,("KEZD+$0B)  adatmennyiség a
                                fejlécből a DE-be
                                CY beállítása 0-ba
        AND     A
        SBC     DE              ha a memóriaterület
        POP     HL              kicsi, akkor E üze-
        CALLC"OERROR"           net
        LD      A,$0D
        MV      DE,"KEZD"+$01
        LD      B,$0B
c4      RST5
        RST4                    programnév kiíratá-
                                sa
        DJNZ    c4              a fejlécről
        RST3
        CP      $4C             az L gomb vizsgála-
                                ta
        JRNZ    $4090           visszatérés a rutin
                                elejére
        MV      ($3E3E),A       képszerkesztés
                                lekapcsolása
c5      CALL    "SINC"          szinkron rutin hívá-
                                sa
        INC     A               ha az azonosító
        JRNZ    c5              nem
                                FFH, akkor újból
        MV      DE,("KEZD"+
                $0B)
        INC     DE              bájtok számának
        LD      C,$FF           beállítása
c6      CALL    "BYTIN"
        LD      HL,A
        XOR     C
        LD      C,A
        INC     HL              adattömb beolvasása
        DEC     DE              és az ellenőrző
        LD      A,D             bájt képzése
        OR      E
        JRNZ    c6
        LD      A,C
        AND     A
        CALL-   "LERROR"        ha az ellenőrző bájt
        NZ                      nem 00H,
                                akkor E üzenet
c14     MV      ($3F3F),A       képszerkesztés
        JR      $4090           visszaállítása
"SINC"
c7      LD      C,$80
c8      CALL    "BYTIN"
        INC     A               keres folyamatosan
        JRNZ    c7              egymás után álló
        DEC     C               FFH bájtokat (hea-
        JRNZ    c8              der)
c9      CALL    "BITIN"         megkeresi az első
        JRC     c9              0 bitet
        CALL    "BYTIN"         beolvassa az
        RET                     azonosító bájtot
"BYTIN"
        LD      A,$01
c10     CALL    "BITIN"         beolvas folyamato-
        RLA                     san
        JRNC    c10             egymás után álló
                                8 bitet és behelyezi
                                az A regiszterbe
        RET
"BITIN"
        PUSH    BC
        EX      AF,AF'
c11     LD      B,"IDŐ"+$80     számláló és a 7. bit
                                beállítása 1-re
        CALL    "ÉL"            két darab élet keres
        CALL    "ÉL"
        JRZ     c11             ha nem talál két
                                élet a ciklusban,
                                akkor újra keres
        EX      AF,AF'
        RL      B               a B regiszter 7.
        CCF                     bitjének komple-
                                mensét
                                beírja a CY-be,
                                ami a beolvasott
                                bitnek felel meg
        POP     BC
        RET
"ÉL"
        MV      A,($3ABF)       vizsgálja az R gomb
        CP      $FB             állapotát
        JRZ     c14
        MV      A,($E000)       a bemenet állapotát
        LD      C,A             beírja C-be
c12     MV      A,($E000)       a bemenetet össze-
        XOR     C               hasonlítja a C regisz-
        RETNZ                   ter tartalmával
        DJNZ    c12
        RET
"LERROR"
        MV      DE,"ERR1"       1E üzenet kezdő-
        JR      c13             címe
                                a DE-be
"OERROR"
        MV      DE,"ERR2"       2E üzenet kezdő-
                                címe
                                a DE-be
c13     RST4                    kiíratás az első
        RETZ                    60H bájtig
        RST5
        JR      c13
```

"KEZD" = szabad tárterület kezdőcíme a RAM-ban
"FREE" = a rendelkezésre álló tárterület mérete
"IDŐ" = időzítő számláló bájt
"ERR1" = 1.hibaüzenet sztringjének kezdőcíme a RAM-ban
"ERR2" = 2.hibaüzenet sztringjének kezdőcíme a RAM-ban
(ERR1) = "←|LOADING ERROR60"
(ERR2) = "←|OUT OF MEMORY60"

```
4090    31 90 40 3E 0D EF DF FE
        4C C2 90 40 32 3E 3E CD
        0C 41 A7 20 FA 21 70 41
        01 00 12 CD 20 41 77 A9
        4F 23 10 F7 32 3F 3F C4
        FF 40 E5 21 7B [3E] ED 5B
        7B 41 A7 ED 52 E1 DC 04
        41 3E 0D 11 71 41 06 0B
        EF E7 10 FC DF FE 4C 20
        B7 32 3E 3E CD 0C 41 3C
        30 FA ED 5B 7B 41 13 0E
        FF CD 20 41 77 A9 4F 23
        1B 7A B3 20 F4 79 A7 C4
        FF 40 32 3F 3F 18 91 11
        4E 41 18 03 11 5D 41 E7
        C8 EF 18 FB 0E 80 CD 20
        41 3C 20 F8 0D 20 F7 CD
        29 41 38 FB CD 20 41 C9
        3E 01 CD 29 41 17 30 FA
        C9 C5 08 06 CA CD 3B 41
        CD 3B 41 28 F6 08 CB 10
        3F C1 C9 3A BF 3A FE FB
        28 B8 3A 00 E0 4F 3A 00
        E0 A9 C0 10 F9 C9 0D 4C
        4F 41 44 49 4E 47 20 45
        52 52 4F 52 60 0D 4F 55
        54 20 4F 46 20 4D 45 4D
        4F 52 59 60
```

Az Aircomp 16 típusú számítógépek 16 k RAM területtel készülnek. Sok megfelelő gyakorlattal rendelkező amatőr kihasználja azt a lehetőséget, hogy ezeket a gépeket kis ráfordítással ki lehet bővíteni 32 k-ra. A rutin 16 k-s géphez készült, de 32 k RAM területtel rendelkező gépen is működik.

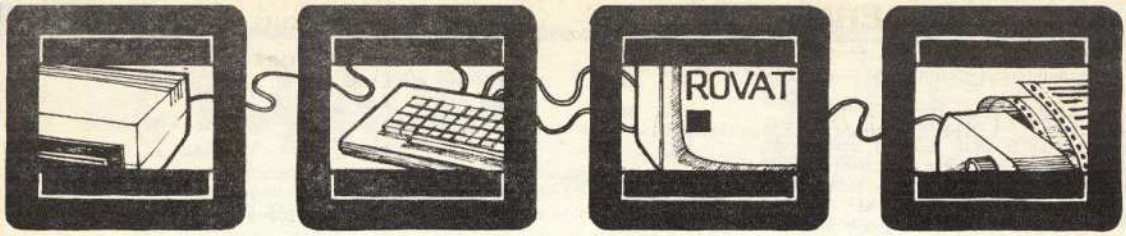
A nagyobb tárterület kihasználására a FREE nevű változót módosítani kell. A gépi kódú listában megjelölt bájtot kell átírni 7E-re, ami a FREE változó magas bájtja.

KANICS MIKLÓS

ELTE Összehasonlító Élettani Tanszéke felvesz

SZOFTVERBEN ÉS HARDVERBEN JÁRTAS VILLAMOSMÉRNÖKÖKET.

Jelentkezés: Budapest VIII., Múzeum krt. 4/a.
Telefon: 189-833/372 mellék.



KARAKTER-GRAFIKA NORMÁL ÜZEMMÓDBAN

Karaktergrafika készítéséhez a C64 grafikus karaktereit használjuk fel normál vagy inverz megjelenítéssel, a lehetséges 16 szín valamelyikére színezve az ábrázolt karaktert. A megtervezett, grafikus és vezérlő karakterekből összeállított ábrát idézőjelek közé téve PRINT utasítással — mint szöveget — közvetlenül is megjeleníthetjük. A cserélhetőség miatt célszerűbb azonban az ábrákat DATA sorokban tárolni, a program elején READ utasításokkal szöveg típusú változókba beolvastatni. Így a különböző változók kiíratásával az ábra megjeleníthető, változtatható, cserélhető, mozgatható.

A program egy mozgó chipet rajzol a képernyőre, melynek mozgása közben a program működését magyarázó szöveget írja ki betűnként a gép. A programot begépelve és futtatva a mozgás egy kicsit darabos, észrevehető a váltás. A szöveg kiírása — a 60-as sor végzi — már lassítja a mozgást, amely tovább fokozható a 70-es sor L ciklusával. Végértékként kb. 50-et írva, a mozgás lassul ugyan, viszont nem olyan bántó az álló képek váltása. A jelenség magyarázata az, hogy gyors mozgásnál egy-egy álló kép „vetítési ideje" kb. azonos a váltás idejével, míg a lassított mozgásnál ez az arány az álló kép javára tolódik el. Így a mozgás lassúbb, egyben azonban gördülékenyebb is lesz.

Képző- és Iparművészeti Szakközépiskola szakköre
ÉNEKES FERENC
tanár vezetésével

```
10 REM ************** EZ A FOPROGRAM *************
20 POKE 53280,0:POKE 53281,0:PRINT "■":F=0:VE=30
30 GOSUB 100 : FOR K=0 TO VE :    REM K=SOROK SZAMA
40 ON F GOTO 60
50 READ SZ$ : IF SZ$="@" THEN F=1 : SZ$= "" : K=VE
60 FOR B=1 TO 40:A$=MID$(SZ$+V1$,B,1):PRINTA$;CH$;
70 FOR C=1TO3:PRINTCH$(C);:FORL=0TO001:NEXTL,C,B,K
80 PRINT"■■■■";:END:REM "■"=CRSR LE + "■"=CTRL+YEL
90 REM ************ BETOLTES SZUBRUTIN ************
100 READ V1$,V$ : V1$=V1$+V1$
110 FOR C=1 TO 3 : READ A$:CH$=CH$+A$+"■"+V$: NEXT
120 FOR C=1 TO 3 : READ A$ : CH$(C)=A$+V$ : NEXT C
130 READ A$ : CH$(3)=CH$(3)+A$ : RETURN
140 DATA"                          ": REM "■"=CTRL + BLK
150 DATA"■■■■■■■■■■■■■■■■■■■■■■■■■":     REM 19 CRSR <-
160 DATA"■ ■  6566 VIDEO   ■" :    REM"■"=INV. ON
170 DATA" ■ INTERFACE CHIP■":    REM"■"=INV. OFF
180 DATA" ■■■■■■■■■■■■■■■■■■■■■■■■■"
190 DATA"  ■■■■■■■■■■■■■■■■■ ":       REM CHR$(180)
200 DATA"  ■■■■■■■■■■■■■■■■■ ":       REM CHR$(194)
210 DATA"  ■■■■■■■■■■■■■■■■■ ":       REM CHR$(170)
220 DATA"■■■■":                  REM 3CRSR↑ + 2CRSR->
225 REM ************ MAGYARAZO SZOVEG ************
230 DATA" COMMODORE 64 KAR.GRAFIKAI PELDAPROGRAM "
240 DATA" KESZITETTE ENEKES FERENC & FOLDI ENDRE "
250 DATA"                                        "
260 DATA"A  VIDEO CHIP  TESTET, ES LABAINAK HAROM"
270 DATA"ALLASAT A 150 - 210 - ES DATA SOROK TAR-"
280 DATA"TALMAZZAK.  A CHIP LEPTETESET A 220 - AS"
290 DATA"DATA SOR VEZERLI.  A  CHIP  RAJZANAK BAL"
300 DATA"OLDALAN  LEVO  ELSO  SZOKOZ  KARAKTER  A"
310 DATA"JOBBRA MOZGATAS UTAN TORLI AZ ELOZO ABRA"
320 DATA"FELESLEGES  RESZET.  A  KIIRASRA  KERULO"
330 DATA"SZOVEG DATA SORAI SORONKENT PONTOSAN  40"
340 DATA"KARAKTERT TARTALMAZZANAK, KULONBEN A KI-"
350 DATA"IRASI KEP ELROMLIK.                     "
360 DATA"A 70 - ES SOR L CIKLUSA A LABAK VALTASAT"
370 DATA"ES EZZEL A CHIP MOZGASAT LASSITJA.  50 -"
380 DATA"ES  VEGERTEKKEL  MAR  GORDULEKENY LESZ A"
390 DATA"MOZGAS. EGYSZERUBB MEGOLDAS A SZOVEG BE-"
400 DATA"TUNKENTI KIIRASA HELYETT EGY ELORE KIIRT"
410 DATA"SZOVEG TORLESE A MOZGO  CHIPPEL.  A 30 -"
420 DATA"AS SOR  CIKLUSANAK  VEGERTEKET  ELERVE A"
430 DATA"CHIP  MEGALL, A PROGRAM FUTASA VEGET ER."
440 DATA"A KIIRANDO SZOVEG VEGJELE A @, SZINTEN A"
450 DATA"PROGRAM  VEGET  JELENTI. A READY FELIRAT"
460 DATA"ES A  VILLOGO  KURZOR AZERT NEM LATSZIK,"
470 DATA"MERT A  TINTA ES A  PAPIR SZINE MEGEGYE-"
480 DATA"ZIK! LASD A 80 - AS SORT!               "
490 DATA"                              FOLYTATJUK!"
500 DATA"@":                            REM END JEL
```

KÉPSZERKESZTŐ

A Megasprite rutin tetszés szerinti nagyságú képet rajzol a képernyő tetszés szerinti helyére. A kép (sprite) adatait megadott helyről veszi.

20—100 — paraméterek betöltése
LINES — a sprite magassága (képsorban megadva)
WIDTH— a sprite szélessége (karakterekben megadva)
ADDR — a képernyő azon címe, ahonnan a rajzolást kezdi
SPADD— az a memóriacím, ahol a képet tároljuk
120—130 — HL és BC értékének elmentése
140—200 — Egy oszlop kirajzolása. Amennyiben az utolsó oszlop is megvolt, vége. Egyébként új oszlop (NEWCOL)
220—280 — HL és BC értékeinek beállítása új oszlop rajzolásához
300—330 — A sprite egy bájtjának megrajzolása
350—360 — Négy bájt munkaterület
380—610 — A szubrutin a HL-ben adott értéket úgy módosítja, hogy az az egy képernyősorral lejjebb lévő címre mutasson
630—680 — A rutin végrehajtása után a rendszer egy gomb megnyomására vár visszatérés előtt. Azonos PAUSE 0.

A rutin a sprite-ot a következőképpen rajzolja. A DE regiszterben megadott helyről a HL által mutatott helyre tölt egy bájtot. HL értékét egy sorral lejjebb helyezi (DOWN), DE-t inkrementálja, majd újra áttölt egy bájtot. Ezt annyiszor ismétli meg, amennyi a magasság (LINES) értéke. Egy oszlop legalsó sora után felülről új oszlopot kezd (NEWCOL). Ez ismétlődik annyiszor, amennyi a szélesség (WIDTH) értéke. A sprite adatait a memóriában tehát oszlopfolytonosan kell elhelyezni.

A rutin gyorsasága érdekében nem ellenőrzi, hogy a sprite nem kerül-e a képernyőn kívülre. Erre vigyázni kell, mert ebben az esetben a rutin a színelemeket, esetleg a rendszerváltozókat fogja átírni.

A MEGASPRITE RUTIN

```
10
20 LINES    EQU 80
30 WIDTH    EQU 8
40 ADDR     EQU 4A30H
50 SPADD    EQU 0000H
60
70          LD, B, LINES          0650
80          LD C, WIDTH           0E08
90          LD HL, ADDR           21 50 49
100         LD DE, SPADD          11 00 C0
110
120 MEGASP  LD (WORKHL), HL       22 9C 8D
130         LD (WORKBC), BC       ED 43 9E 8D
140 COLUMN  CALL DRAW             CD 98 8D
150         CALL DOWN             CD A0 8D
160         DJNZ COLUMN           10 F8
170         DEC C                 0D
180         JP Z, END             CA D5 8D
190         CALL NEWCOL           CD 8A 8D
200         JR COLUMN             18 EF
210
220 NEWCOL  LD HL, (WORKHL)       2A 9C 8D
230         INC HL                23
240         LD (WORKHL), HL       22 9C 8D
250         LD A, C               79
260         LD BC, (WORKBC)       ED 4B 9E 8D
270         LD C, A               4F
280         RET                   C9
290
300 DRAW    LD A (DE)             1A
310         LD (HL), A            77
320         INC DE                13
330         RET                   C9
340
350 WORKHL  DEFW 0000             0000
360 WORKBC  DEFW 0000             0000
370
380 DOWN    XOR A                 AF
390         OR A                  B7
400         BIT 3, H              CB 5C
410         JR NZ, DOWN1          20 E
420         INC H                 24
430         BIT 3, H              CB 5C
440         RET Z                 C8
450         RES 3, H              CB 9C
460         LD A, L               7D
470         ADC A, 20H            CE 20
480         LD L, A               6F
490         RET NC                D0
500 DOWN2   SET 3, H              CB DC
510         RET                   C9
520 DOWN1   INC H                 24
530         BIT 4, H              CB 64
540         RET Z                 C8
550         RES 4, H              CB A4
560         LD A, L               7D
570         ADCA, 20H             CE 20
580         LD L, A               6F
590         JR NC, DOWN2          30F1
600         SET 4, H              CB E4
610         RET                   C9
620
630 END     LD HL, 23560          21 08 5C
640         LD (HL), 0            3600
650         LD, A, 0              3E 00
660 NPRESS  CP (HL)               BE
670         JR NC, NPRESS         30 FD
680         RET                   C9
690
```

MAKAI MÁTYÁS—SCHMITT PÁL
Közgazdaságtudományi Egyetem

APRÓ FOGÁS

Spectrum számítógépemhez EPSON nyomtatót használok. Listázáskor a sorok rendezetlenek, áttekinthetetlenek lesznek, mivel hosszúságuk különböző. További hiba, hogy az RX—80 nem tudja karakter üzemmódban a grafikus karaktereket kinyomtatni. Ezek kiküszöbölésére találtam ki a következő módszert, ami rövidebb programokhoz használható. A beolvasás után beállítom a futást, és beírom a LIST parancsot. A „scroll?" feliratra BREAK-et nyomok, majd beírom a COPY parancsot. Ezután újra listáztatok elölről, de a második képernyő után COPY-zok, és így tovább. Hosszabb programoknál ez a módszer kényelmetlen, de néha megéri a fáradságot.

RUSZNYÁK GÁBOR
Kolos Richárd Szakközépiskola

SZÁMÍTÓGÉP-FEJLESZTŐK! GYÁRTÓK!

Nálunk azonnal kapható MOM 1800/900 típusú 5¼" hajlékonylemezes meghajtóegység. Tápegység és doboz nélkül ára 18 000,— Ft.

Érdeklődni lehet a 252-880-as telefonszámon

A SZÁMÍTÓGÉP ANATÓMIÁJA

Ha ma Magyarországon valaki számítástechnikával akar foglalkozni, először a programozással kezd ismerkedni. Amikor ebben elért egy bizonyos szintet, és még mindig nem veszítette el a lelkesedését, valószínűleg érdeklődni kezd a gépi programozás iránt. Eközben megismeri a számítógép felépítését programozási szempontból. De mivel senkitől nem kap segítséget a hardver megismeréséhez, többnyire tovább nem is jut.

Ennek több hátránya is van. Egyrészt nem tudja kihasználni a gép hardverjéből adódó lehetőségeket, másrészt nem tudja saját maga bővíteni rendszerét. Sok érdeklődőnek ezért nincs saját számítógépe: nincs elég pénze, hogy vásároljon, építésre pedig gondolni sem mer.

Ezen a hiányosságon szeretnénk segíteni cikksorozatunkkal. Hogy mindenki számára érthető legyen a téma, két oldalról közelítjük meg: egyrészt az elektronika szempontjából, másrészt a programozás felől.

Nézzük meg először, milyen alapvető egységekből áll a számítógép? Mindenekelőtt tartalmaznia kell egy központi egységet (central processing unit — CPU). Ez hajtatja végre az utasításokat és vezérli a rendszer működését. Hogy ezt miként teszi, azzal később részletesen foglalkozunk.

Az utasításokat és az adatok nagy részét a memória tartalmazza. A külvilággal való kapcsolatot a perifériák biztosítják. A perifériák egyik fajtája a nagy tömegű, olcsó adattárolásra szolgál. Ezeket háttértáraknak nevezzük.

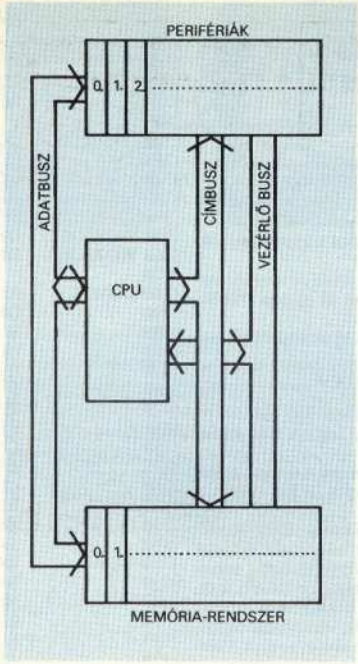
Mint azt a programozás közben megtanulhattuk, a tár és a periféria is számozott rekeszekből áll. Minden ilyen rekeszbe egy, a számítógépre jellemző hosszúságú számot írhatunk és/vagy olvashatunk. A memória eltárolja ezt a számot, a periféria pedig a legkülönfélébb dolgokat csinálhatja vele: például kinyomtatja, elhelyezi a háttértárolón vagy saját működését befolyásoló parancsként értelmezi.

Az egységeket egy három jól elkülöníthető részből álló vezetékrendszer, a busz vagy más néven sín kapcsolja össze. Ez a három rész a cím-, az adat- és a vezérlőbusz. Hogy működésüket megértsük, készítsünk modellt a számítógép működésének illusztrálására! *(1. ábra)*

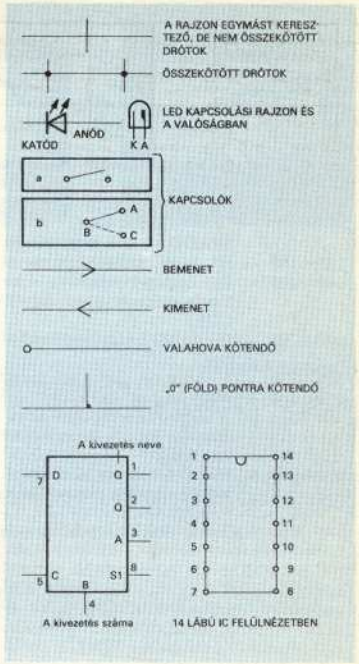
A központi egység ideje jelentős részét azzal tölti, hogy a memória- vagy perifériarekeszek egyikét írja vagy olvassa. Hogy éppen hányadik rekeszt olvassa, azt a memória vagy a periféria a címbuszról tudja meg. A címbuszon ez a szám van bináris (kettes számrendszerbeli) formában, elektromos jelek alakjában. A bináris szám minden helyiértékének megfelel a címbusz egy vezetéke. Ha a vezetéken HI szint van (lásd lejjebb), akkor ez a számjegy 1, ha LO szint, akkor Ø. Az adat, amit a központi egység ír vagy olvas, ugyanígy kódolva, az adatbuszon van.

Hogy a központi egység ír vagy olvas-e, hogy perifériához vagy memóriához fordul-e, valamint egyéb állapotai a vezérlőbuszról olvashatók le. Erről részletesebben is lesz még szó.

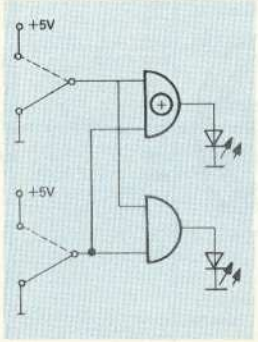
Mit látunk, ha szétszedünk egy számítógépet? Egy műanyag lapot, tele elektronikus alkatrészekkel. Az alkatrészek jó része kicsi, fekete vagy barna, és temérdek lába van. Egy ilyen tokban egy-egy komplett áramkör található, amelyet integrált áramkörnek (integrated circuit — IC) nevezünk.



1. ábra



2. ábra



3. ábra

BEMENETEK ÁLLAPOTA		ÉS	VAGY	KIZÁRÓ VAGY	NEMÉS	NEM-VAGY	NEM*
		AND	OR	XOR	NAND	NOR	INVERTER
A	B						
L	L	L	L	L	H	H	H
L	H	L	H	H	H	L	H
H	L	L	H	H	H	L	L
H	H	H	H	L	L	L	L

*Csak A bemenete van

4. ábra

A számítógép IC-jének bemente csak kétféle jelszintet különböztet meg. (Ezt nevezzük digitális technológiának, szemben az erősítő stb. analóg technológiájával.) Az egyik általában 0—0,8 V közötti, ez az ún. alacsony szint (low, röviden LO). A másik 2,4—5 V-ig terjed, ez a magas (high, HI) szint. Ha a bemeneten 0,8 és 2,4 V közötti szint van, az IC nem biztos, hogy helyesen fog működni. Az itt leírt szintek a leggyakrabban használt logikai áramkörcsaládra, a TTL és TTL-kompatibilis IC-kre vonatkoznak, ahol a tápfeszültség 5 V ±0,5 V. Ennél jóval modernebb a CMOS család, amely sokkal kevesebbet fogyaszt. Tápfeszültsége 3 és 15 V között bármekkora lehet.

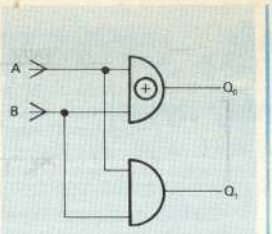
Ha egy rádiót vagy egy erősítőt szétszedünk, szintén találhatunk benne IC-t. Mi a különbség a kettő között?

Az erősítő IC-ben a bejövő és kimenő feszültség bizonyos határok között bármilyen értéket felvehet, és minden különböző bejövő értékhez különböző kimenő érték tartozik. Például egy erősítőnél ez a jelszint a hangerőt jelenti, ami folyamatosan változhat.

Többször használtuk már a feszültség szót. Ez az elektronikában leggyakrabban használt fizikai mennyiség neve. U-val jelöljük. Feszültség mindig két pont között mérhető, és azt a munkát jelenti, amellyel az egységnyi elektromos töltésnek az egyik pontról a másikra való átjutása jár. A gyakorlatban, ha egy bizonyos pontnak a feszültségszintjét (potenciálját) adjuk meg, az a tápfeszültség 0 voltjához viszonyított fe-

A KÉT EGYBITES SZÁM BEMENETE		Q_1	Q_0	AZ ÁLLAPOT ARITMETIKAI JELENTÉSE
A	B			
L	L	L	L	0+0=0
L	H	L	H	0+1=1
H	L	L	H	1+0=1
H	H	H	L	1+1=2 (binárisan 10)

5. ábra



6. ábra

szültséget jelenti. Ezt is U-val jelölik. A tápfeszültség mindig a valamilyen áramköri részlet működéséhez szükséges energiát szolgáltatja.

Az elektronikus készülékek felépítését legjobban a kapcsolási rajz írja le. Ennek számunkra most legfontosabb jelöléseit (a logikai kapuk kivételével) a *2. ábra* tartalmazza. Ezek közül kettő magyarázatra szorul. Az egyik a LED, más néven világító dióda. Ez egy olyan alkatrész, amely feszültség hatására világít valamilyen nem fehér színnel. Sokkal jobb a hatásfoka és nagyobb az élettartama, mint a villanykörtéé, mert nem termel hőt. A másik magyarázandó alkatrész a kapcsoló. Az a) típus egyik állásban összeköti a két kivezetését, a másikban nem. A b) típus egyik állásban A-t és B-t, a másikban B-t és C-t köti össze. A kapcsolási rajzra példaként szereplő *3. ábrát* a cikk végigolvasása után meg fogjuk érteni, és rá fogunk jönni, mire használható ez a kapcsolás.

A logikai szinteknek nemcsak számokat, hanem logikai értékeket (igaz, hamis) is megfeleltethetünk. Ezekkel pedig elvégezhetjük a programozásból jól ismert logikai műveleteket (ÉS, VAGY, NEM stb.). Azok az áramkörök, amelyek ezeket a műveleteket elvégzik, a logikai kapuk. A számítógépek elsősorban ilyen kapukból épülnek fel. Egy-egy bonyolultabb IC-ben néhány ezer kapu is lehet. Természetesen ezek közül csak néhány van közvetlen kapcsolatban a külvilággal.

A logikai kapuk jelölésére többféle szabvány létezik. Ezek közül a legelterjedtebb jelöléseit a *4. ábrán* láthatjuk, igazságtáblázatukkal együtt. AND, OR, NAND és NOR kapuk lehetnek több, mint kétbemenetűek is. Értelemszerűen a sokbemenetű AND kapunál is csak akkor HI a kimenet, ha az összes bemenet HI szinten van, az OR-nál pedig csak akkor LO a kimenet, ha minden bemenet LO.

A digitális áramkörök működését legrészletesebben az ún. igazságtáblával lehet jellemezni. Ez a táblázat azt mutatja, hogy milyen bemeneti értékekhez milyen kimeneti állapot tartozik.

Egy egyszerű logikai áramkör tervezésénél az első feladat az igazságtábla elkészítése. Legyen például a feladat egy egybites számokat összeadó berendezés készítése, ami az eredményt két biten ábrázolja. Itt, mint általában az aritmetikai áramkörökben, a HI szint jelenti az 1-et, a LO pedig a 0-t.

Írjuk fel az áramkör igazságtábláját *(5. ábra)*! A táblázatban A és B bemenetek, Q_0 a kimenet $2^0=1$ helyiértékű, Q_1 a $2^1=2$ helyiértékű számjegye. Ha összevetjük a Q_0 kimenet A és B bemenetekről való függését a logikai kapuk igazságtáblájával, azt találjuk, hogy pontosan megegyezik a XOR kapuéval. Hasonlóképpen a Q_1 kimenet táblája az AND kapuéval egyezik meg.

Ezek után elkészíthetjük az áramkör tervét *(6. ábra)*. A számítógép aritmetikai egységében ehhez hasonló áramkörök végzik az összeadást.

Gyakorlásul tervezzünk kivonó áramkört! Az eredményt kétbites kettes komplemens formában adja. Ez azt jelenti, hogy a −1-et 11-nek jelöljük a kimeneten. A megoldást a következő számban közöljük.

Miután megterveztük a berendezést, meg kell határozni, hogy milyen IC-k szükségesek hozzá. Ezt valamelyik forgalomban levő katalógus alapján végezhetjük el. Esetünkben egy 7408 és egy 7486 típusú IC kell. Ezek 4-4 kaput tartalmaznak. Azt, hogy az IC melyik lába (kivezetése) melyik kapuhoz tartozik, szintén a katalógusból tudhatjuk meg.

A következő lépés az áramköri lemez (nyomtatott áramkör — NYÁK) megtervezése, elkészítése, majd az alkatrészek beültetése. Ezzel a témával sokat foglalkoznak különféle népszerűsítő könyvek és például a Rádiótechnika című újság, itt nem részletezzük.

Vigyázat! A NYÁK gondatlan megtervezésével vagy hibás beültetésével komoly károkat okozhatunk! Egy fordítva behelyezett IC többnyire tönkremegy.

A következő részben többek között arról lesz szó, hogy miként lehet kapukból a számítógép alapvető áramköreit előállítani.

FEHÉR TAMÁS—MOLNÁR ATTILA
Berzsenyi Dániel Gimnázium

BILLENTYŰZETFELJAVÍTÁS

Aki már dolgozott kissé használt iskolaszámítógépen, az bizonyosan tapasztalta, hogy a sokat használt billentyűk néha — ez a „néha" esetleg tíz esetből kilenc is lehet — dupláznak, és a képernyőn nem egy, hanem két betű jelenik meg. Ez a kellemetlentől egészen a használhatatlanságig fokozódhat, hiszen sokszor a visszalépő billentyű is dupláz. A jelenség oka a HT „nem egészen tökéletes" billentyűzete és a billentyűvizsgálat gyorsasága — az egyetlen funkciója a gépnek, amit eléggé gyorsan hajt végre. A billentyű-ellenőrző rutin ugyanis egy billentyű lenyomására azt érzékeli, s ha még idejében visszakapja a vezérlést, akkor a billentyűt még mindig lenyomva találja, tehát azt még egyszer érzékeli. Valójában a probléma a billentyűzetben levő kapcsolók számlájára írható.

A megoldás a magyarázatban rejlik. Várakoztassuk meg a rutint néhány tizedmásodperccel, hogy a billentyűnek legyen ideje felemelkedni. Írjuk be tehát a következő gépi kódú programot az EDI segítségével:

```
 1 ORG 4016H
 2 LOAD 4016H
 3 DW 4152H
 4 ORG 4152H
 5 LOAD 4152H
 6 PUSH AF
 7 PUSH HL
 8 LD HL,0201H
 9 CIM:DEC HL
10 LD A,H
11 OR L
12 CP 0
13 JR NZ,CIM
14 POP HL
15 POP AF
16 JP 03E3H
17 END
```

A 4016H címen a gép a billentyűvizsgáló rutin címét tárolja két bájton. Ide kell beírnunk a várakozó program kezdőcímét. Ez jelen esetben 4152H — a lemezes rendszer parancsterülete. Természetesen máshova is írhatjuk, de gondoskodnunk kell a terület védelméről.

A várakozás igen egyszerű: a HL értékét csökkentjük nulláig. Azért nem DJNZ-t használunk, mert az túlságosan kevés lenne. Ezután a vezérlést vissza kell adni az eredeti programnak, ami itt JP 03E3H. Nem szabad azonban a regiszterek értékét megváltoztatni, ezért az általunk használt regisztereket a veremtárba kell elmenteni. Fordítsuk le EDI-vel a programot, de az OPTION? kérdésre ne írjunk semmit, mert gépünk elszáll. Ezután – ha mindent jól csináltunk – többet nem dupláz a billentyűzet.

Vigyázat! Nehogy eszünkbe jusson futtatni! Vegyük fel ezután kazettára (4016H-tól kb. 4170H-ig), de kezdőcímnek 1A19H-t írjunk, hogy beolvasáskor BASIC-be tudjunk visszatérni.

A várakozás hosszát a POKE16726,X utasítással állíthatjuk be gépünk állapotának megfelelően. Az X lehetőleg nullától tízig terjedjen, ellenkező esetben a gép esetleg másodperceket vár egy-egy billentyű elfogadására.

Hasznos lehet a POKE16726,0 BASIC programok elején, mert a gép minden utasítás végrehajtása után megvizsgálja a billentyűzetet, és a várakozás jelentősen lelassítaná a futást. INPUT előtt és kiszállás után azonban vissza kell írni a megfelelő értéket. A BASIC-bővítések általában megszüntetik a program hatását, mert visszaírják az eredeti kezdőcímet, a RESET gomb viszont szerencsére nem nyúl hozzá.

PALLAGI LÁSZLÓ
Vörösmarty Gimnázium Érd

Assemblerek, cross-assemblerek

2. Az Intel 8080 assembly nyelv

A (cross-)assemblerek működését egy valódi assembly nyelv segítségével mutatjuk be. Erre önként adódik az Intel cég 8080-as mikroprocesszorának nyelve, hiszen az Intel az egyik legnagyobb hagyományokkal és legkiforrottabb, legszélesebb körben használt mikroprocesszor-családokkal rendelkező gyártó cég. A 8080 típusú mikroprocesszor egyik legismertebb termékük, amelyet ugyan már sok éve gyártanak, és ezért nem tekinthető a legkorszerűbbnek, mégis bizonyára Magyarországon is erre készült a legtöbb fordítóprogram.

Kötöttségek és lehetőségek tervezéskor

Az egy konkrét típushoz való kötődés korántsem megy az általánosság rovására. Hiszen a fordítóprogram elkészítéséhez elegendő, ha a (cross-)assembler tervezője magának a 8080 mikroprocesszornak a felépítéséről, működéséről csak néhány olyan, az utasításkészlettel kapcsolatos részletet ismer, mint a tárgykód felépítése, a címzési módok, a regiszter és regiszterpár típusú utasítások és ezek megadásának, valamint a tárgykódba való beépítésüknek módja, közvetlen operandusú utasítások, amelyek az operandust nem egy tárcímről vagy egy regiszterből, hanem magából az utasítás tárgykódjából olvassák ki stb.

E kötöttségek betartásán kívül a felhasználó, ha maga írja a fordítóprogramot, viszonylag szabadon kezelheti az Intel által definiált nyelv szabályait: bizonyos korlátok — amelyekre részben a későbbiekben még visszatérünk — között tetszőlegesen szűkítheti vagy bővítheti a formai előírásokat (a nyelv szintaktikáját), illetve a nyelv tartalmi adottságait, lehetőségeit (szemantikáját).

Megjegyezzük, hogy a 8080 processzor nyolcbites architektúrája sem jelent a lényeget érintő korlátozást, mert az újabb, 16 és 32 bites típusokhoz készített fordítóprogramok elvileg mit sem különböznek az itt bemutatásra kerülőtől.

A forrásnyelvi sor mezői

A gyakorlatban használt, assembly nyelveken írt programokban minden utasítás külön sorban foglal helyet. Az egyes utasításssorok mezőkre oszlanak. A mezők elhelyezkedése szempontjából a nyelv kötött vagy szabad formátumú lehet. Kötött formátum esetén az egyes mezők a forrásnyelvi sornak csak előre kijelölt sorszámú pozícióit foglalhatják el. Ezzel szemben szabad formátum esetén az egyes mezőket mindössze egy vagy több ún. üres karakter (szóköz) választja el egymástól. Az egyszerűség kedvéért feltételezzük, hogy a vizsgált nyelvben kötött formátumú utasításssorokat

Nyolcrészes sorozatunk a mikroszámítógépek assemblereiről, cross-assemblereiről szól. Célja, hogy a cross-assemblerek példáján keresztül megismertesse az olvasót az assembler programok működésével. A bemutatáshoz a rendszermodellezési eszközöket használjuk fel, s így készítjük el az assemblerek működésének egy szabványosított algoritmusát, modelljét. E cikkünk a sorozat második része.

1. (Cross-)assemblerek és a rendszermodellezés
2. Az Intel 8080 assembly nyelv
3. A (cross-)assembler program mint rendszer
4. A rendszer működése I. — A fordítás két menete
5. A rendszer működése II. — Táblák és adatterületek
6. Az operátorkészlet I.
7. Az operátorkészlet II. — A rendszer kapcsolatábrája
8. Példák a rendszer működésére.

használunk. A sorok a következő öt mezőből állnak.

Kontrollmező. Ha tartalma nem szóköz, akkor az egész utasítássor megjegyzésnek veendő.

Címkemező. Címke vagy azonosító definiálására szolgál. Címkének tekintjük az olyan szimbolikus nevet, amely a hozzá tartozó gépi utasítás tárban való elhelyezkedésének a címére mutat. Ez a tárcím a címke értéke. Az azonosító olyan szimbolikus név, amely értékét a programban elfoglalt helyzetétől függetlenül, direktívával való értékadás (lásd később) révén kapja.

Műveletikód-mező. Egy gépi utasítást, egy direktíva mnemonikus kódját vagy egy makrónevet tartalmaz. Lényege, hogy bármelyik esetben csak előre rögzített készletben szereplő, és ezért az érvényesség szempontjából ellenőrizhető karaktersorozat jelenhet meg a mezőben.

Operandusmező. Tartalmát aszerint kell értelmezni, hogy a műveletikód-mezőben mi szerepel. A gépi utasításhoz tartozó operandusmező olyan kisegítő információkat ad meg, amelyek az utasítás műveleti kódját egészítik ki, s így e kettő együttesen határozza meg pontosan a végrehajtandó utasítást. Ebben az esetben a mező négyfajta információt szolgáltathat: a műveletben részt vevő (1) regiszter vagy (2) regiszterpár nevét, (3) a közvetlen operandust vagy (4) a tárban tárolt operandus címét. Direktíva esetén az operandusmező feladata, hogy közvetlenül a fordítóprogramnak adjon a fordítás menetét szabályozó információt. Ha makrónév áll a műveletikód-mezőben, akkor az operandusmező hívási paramétereket (lásd alább) adhat meg.

Megjegyzésmező. Az operandusmezőtől egy kitüntetett karakter, többnyire legalább egy szóköz választja el. A programozó tetszőleges információt elhelyezhet a mezőben, amit a fordítóprogram figyelmen kívül fog hagyni.

Az egyes utasításokban nem kötelező minden mezőt kitölteni. Ez nyilvánvaló a kontroll- és a megjegyzésmezőre vonatkozóan. Ha azonban a kontrollmező üres, akkor a műveletikód-mezőben érvényes műveletnek (gépi utasításnak, direktívának, makrónévnek) kell szerepelnie. A kijelölt művelet jellege határozza meg, hogy a címke- és az operandusmezőket kötelező-e kitölteni, vagy a programozó — tetszés szerint — akár üresen hagyja őket, akár (az operandusmező esetében) ír oda valamit, az a fordítóprogram számára közömbös.

A direktíva

Minden assembly nyelv fontos alkotórészei a direktívák. Ezek nem a program futása során végrehajtandó utasításokat határoznak meg, hanem a fordítóprogram saját maga számára értelmezi és hajtja végre őket. E direktívák a fordítóprogram működését szabályozzák, és a fordítás során keletkező tárgyprogramban többnyire nincsen nyomuk, noha vannak olyanok is, elsősorban az adatmezők definiálására szolgálók, amelyek tárgykódot is létrehoznak.

A direktívák sohasem az adott mikroprocesszortól, illetve a számítógép központi egységétől függenek, vagyis nem rájuk, hanem kizárólag az assembly nyelvre, illetve a megvalósított assembler fordítóprogramra jellemzők. Vannak köztük olyanok, amelyek minden assembly nyelvben megjelennek, többek között éppen a fent említett, az adatmezők definiálását végző direktívák, és vannak olyanok is, amelyek a programozói munka megkönnyítése érdekében, többletszolgáltatásként kerülhetnek be a fordítóprogramba.

Direktívatípusok

Az egyes assemblerekben megvalósított direktívák, ha egyedi megjelenésükben különböznek is egymástól, mégis néhány jól körülhatárolható típusba sorolhatók. Ezek közül csak azokat mutatjuk be, amelyeket az általunk tárgyalt (cross-)assembler program is tartalmaz.

A fordítási kezdőcím kijelölésére szolgá-

ló direktívatípus tudatja a fordítóprogrammal, hogy a forrásprogramot, illetve annak egy modulját mely tárcímtől kezdődően kívánjuk fordítani. Így ez a direktívatípus adja meg az ún. beültetési számláló kezdeti értékét. A beültetési számláló pillanatnyi értéke mindig azt mutatja meg, hogy a soron következő utasítás vagy esetleg direktíva tárgykódját mely tárcímtől kezdődően kell elhelyezni. Fordítóprogramunkban az *ORG* nevű direktíva állítja be a számláló kezdeti értékét.

A forrásprogram végét jelző direktívatípus közli a fordítóprogrammal, hogy a forrásprogram, illetve egy modulja befejeződött. Ennek a direktívának az operandusa szolgál általában arra, hogy a programfutás első végrehajtandó utasítását, a programindítás belépési pontját megadja. Esetünkben *END* e direkíva neve.

Az adatmezők definiálását végző direktívatípus részben a programban szükséges állandók, részben a műveletek eredményeit tároló vagy munka-, illetve pufferterületként használt mezők definiálását végzi, ami térfoglalással is jár. Az Intel nyelv három ilyen típusú direktívát ismer: a *DB* direktíva 1 bájtos (8 bites) adat vagy 1 bájtos adatokból álló lánc (állandó, illetve állandók) tárolására, a *DW* direktíva 2 bájtos adat vagy 2 bájtos adatokból álló lánc (állandó, illetve állandók) tárolására (mégpedig a 2 bájton belül az alacsonyabb és a magasabb helyiértékű bájtok felcserélve), a *DS* direktíva pedig az operandusmezőben megadott számú bájtok lefoglalására alkalmas.

Az azonosítónak értéket adó direktívatípus arra szolgál, hogy — mint már láttuk — az azonosító értékét, a címkétől eltérően, a programban való megjelenési helyétől függetlenül adjuk meg. Megjegyzendő, hogy az azonosító csak a fordítás idejére érvényes változó, tárfoglalás vele kapcsolatban nem történik. Fordítóprogramunk a *SET* és az *EQU* direktívákat ismeri ebből a típusból. Mindkettő hatására a címkemezőben megjelenő azonosítónévhez az operandusmezőben szereplő kifejezés értéke rendelődik hozzá. Különbség, hogy az *EQU* direktívával csak egyszer, a *SET* direktívával többször is értéket lehet adni ugyanannak az azonosítónak. Egy további eltérésre a makrócímkék tárgyalásakor térünk vissza.

A fordítóprogramok sok esetben lehetővé teszik, hogy bizonyos forrásprogramrészletek fordítására ne kerüljön sor, ha a fordítóprogram erre külön „felszólítást" kap. Ekkor az adott programrészletet a fordítóprogram átugorja, megjegyzésként tekintve a benne szereplő sorokat. Ha viszont a fordító nem kap utasítást a kihagyásra, akkor a forrásnyelvi sorokat szabályosan feldolgozza.

A feltételes fordítással kapcsolatos direktívák feladata az, hogy egyrészt kijelöljék a feltételes fordítás hatálya alá eső programrészleteket, másrészt vezéreljék magát a fordítást is. A kijelölés egy mindig párosan fellépő direktívakettőssel, esetünkben az *IF* és az *ENDIF* direktívákkal történik, amelyek két oldalról határolják az illető programrészletet. A fordítás vezérlésére felhasználható az elöl álló (*IF*) direktíva: ha ennek az operandusmezejében szereplő kifejezés értéke Ø, akkor a záródirektíváig (*ENDIF*) terjedő sorok kimaradnak a fordításból. A direktívapárokat tetszőleges mélységben egymásba is lehet skatulyázni.

A makrók definiálására szolgáló direktívatípus feladata egy ugyancsak kitüntetett módon kezelt forrásprogramrészletnek, az ún. makrónak a definiálása. A forrásprogramrészlet a mindig párosan alkalmazott *MACRO* és *ENDM* direktívák közé van zárva.

Jelentőségük miatt a makrókkal kicsit részletesebben is foglalkoznunk kell.

A makrók

A makró olyan, forrásnyelvén írt s külön tárolt (vagy tárolásra kerülő) programrészlet, amely a felhasználói programban tetszőleges helyen ún. makróhívás segítségével beiktatható. A hívás helyén, a makróban definiált programrészlet (az ún. makrótörzs) bemásolódik a forrásprogramba. A makró rugalmasabb alkalmazása érdekében lehetőség van arra is, hogy az egyes hívásoknál a bemásolási hely szövegösszefüggéséhez igazítva történjék a beiktatás. Erre részben a makrók paraméterezhetősége, részben a korábban tárgyalt feltételes fordíthatóság nyújt módot. A paraméterezhetőség a formális-aktuális paraméterek cseréjének ismert módú alkalmazását jelenti.

A makrókezelés feladatai

A makrók kezelésével kapcsolatban háromféle tevékenységet kell megkülönböztetni: a makró definiálását, a makróhívás felismerését és a makrókifejtést, vagyis a makrótörzs bemásolását a forrásprogramba.

A makró definiálása a *MACRO* és *MEND* direktívák közé zárt assembly nyelven írt utasításssorok megadásával történik. A *MACRO* direktíva sorában lévő címkemező tartalma a makró nevét határozza meg, az operandusmező pedig az esetleges formális paramétereket adja meg. A direktíva hatására a fordítóprogram ideiglenesen, vagy szükség esetén az ún. makrókönyvtárban állandó megőrzésre tárolja a *MEND* direktíváig terjedő programrészletet. A tárolásról bejegyzés készül egy ún. makródefiníciós táblába, s ezzel a makródefiniálással összefüggő tevékenység befejeződik.

Ideiglenes tárolás esetén abból a programból, ahol a makró definiálásra került, állandó tárolás esetén pedig tetszőleges programból hívható a makró. A hívás módja: a program szükséges helyén a műveletikód-mezőben a makró nevét kell szerepeltetni. A fordítóprogramnak fel kell tudnia ismerni, hogy se nem utasítás, se nem direktíva, hanem egy már korábban definiált makró szerepel a mezőben. Ha a hivatkozott makró definíciójában voltak formális paraméterek, akkor a hívósor operandusmezeje egy ún. paraméterlistát tartalmazhat. A paraméterlista elemei az aktuális paraméterek.

A hívás felismerése után következik a makró kifejtése: a fordítóprogram soronként „előveszi" a tárolt makródefiníciót, elvégzi a formális-aktuális paraméterek cseréjét, s az így „aktualizált" makrótörzset bemásolja a hivatkozás helyén. A makródefiníció utolsó sorának bemásolása után a fordítóprogram visszatér az eredeti program következő sorának feldolgozásához.

Fordítóprogramunkban megengedjük a makróhívások egymásba skatulyázását, azaz egy makrótörzsben is lehet egy (másik) makróra hivatkozni. Ebben az esetben a makrókifejtés során végzendő műveletek elvileg változatlanok a fentiekhez képest, a megvalósítás viszont némileg bonyolultabbá válik.

Címkekezelés a makróban

Befejezésül külön kell szólni a makrókon belüli címkekezelésről. A makrótörzsben tetszőleges, szintaktikailag helyes forrásnyelvi utasítássorok előfordulhatnak, így az utasításoknak, direktíváknak, makróhivatkozásoknak lehetnek címkéi is. A makró definíciójában szereplő címkét a fordítóprogram lokális címként kezeli, amelynek érvényessége csak a makrótörzsre korlátozódik. Lehet mód azonban arra is, hogy makrón belül globális — tehát az egész programra, illetve programmodulra kiterjedő érvényességű — címkét is definiáljunk, de ekkor a makrót a programban, illetve a modulban csak egyszer szabad meghívni.

Az elmondottak vonatkoznak az *EQU* direktívával értéket nyert azonosítókra is. Makrón belül *EQU*-val (nem globálisnak) definiált azonosító csak a makrótörzsön belül érhető el. Más a helyzet a *SET* direktívával értéket kapott azonosítókkal. A makrótörzsben szereplő SET direktíva nemcsak egy lokális azonosítónak ad értéket, hanem új értéket adhat az ugyanilyen nevű modulbeli azonosítónak is, ha egyáltalán van ilyen.

Ezzel röviden áttekintettük az Intel 8080 mikroprocesszor assembly nyelvének sajátosságait. Egyes kérdésekre a későbbiekben még részletesebben vissza kell térnünk, amikor a (cross-)assembler által elvégzendő feladatokat és a megvalósítási módjukat írjuk le. Előbb azonban meg kell alapoznunk a fordítóprogramok rendszerszemléletű tárgyalását, s ehhez néhány fogalmat és alapelvet kell tisztáznunk. A következő részben tehát a (cross-)assembler programmal, mint rendszerrel foglalkozunk.

VÁRGEDŐ TAMÁS

Kedves Olvasóink!

Olyan kéziratokat, amelyeket nem rendeltünk meg, nem őrzünk meg és nem küldünk vissza. Levelek, cikkek stb. közlése esetén szerkesztőségünk fenntartja a jogot az írások rövidítésére.

Újszerű közművelődési informatizálási törekvéseinkkel már találkozhattak a magazin előző számában is. Az ott közölt pályázati felhivást mi adtuk közre, a személyi számítógépek közművelődési, humánus és rentábilis alkalmazásának segítésére. Felhívásunkat fenntartjuk. Várjuk továbbra is a pályázati jelentkezéseket.
A székesfehérvári nyilvános megbeszélés időpontja módosult: 1986. ápr. 22-26.
Címünk, telefonunk változatlan: Népművelési Intézet TTM Csoport Bp 1251 Pf 33 Corvin tér 8.
T:388-122/34,70; 388-176.

VÉDJEGYVÁLLALKOZÁSI KONCEPCIÓ

Az első közművelődési védjegy-üzemeltető vállalkozás elvi felvetése

Az eddigi hazai számítógép terjesztési programok a "lakosságot" csak mint programozási tanfolyamok hallgatóit vették számításba. Csupán a legutóbbi hónapokban merült fel az ennél szélesebben is értelmezhető "lakossági informatika" egyelőre bizonytalan jelentésű fogalma. A mi informatizálási törekvésünk a számítógép-alkalmazás teljesebb problémakörét veszi alapul, és ezért a személyi számítógépek üzemeltetésének rentabilitását fontos követelményként hangsúlyozza a művelődési házak esetében is. Sőt. Külön kiemeljük ki a személyi számítógépekkel végezhető szolgáltatások megfizethetőségét az átlagos háztartás részéről. A háztartásoknak szánt szolgáltatások kifejlesztéséhez a pályázati kerettel, a már kialakult szolgáltatások folyamatos, színvonalas működtetéséhez pedig a következőkben felvetett védjegyvállalkozással kívánunk szerény mértékben hozzájárulni.

I.A védjegyvállalkozás ötletének eredete

Számunkra, mint a Népművelési Intézet egyik új csoportjának a következő feladat jutott: segíteni kellene a művelődési házakat a számítógépek sikeres alkalmazásában. De hogyan?

Itt, most a szemléletesség kedvéért két, alapvetően különböző eljárást állítunk szembe egymással:

A/ Központi kampányokat lehet szervezni (preferált felszerelések, géptípusok szétosztására, vetélkedőkre stb), a fejleszteni kívánt intézményeket mintegy versenyeztetve a "szűkösen könnyű" központi pénzekért. Ehhez központi pénzek kellenek egyrészt (amivel csoportunk nem bővelkedik), másrészt ez a módszer nem teszi lehetővé a helyi erőforrások katalizálását, amint az közgazdászok előtt ma már közismert.

B/ A helyi erőforrásokat, ambíciókat egy alapvetően más, újszerű (és időszerű) eljárással lehetne mozgósítani, eredményessé tenni. Ez esetben sokkal inkább figyelembe vehetők a leendő számítógéphasználók majdani problémái, lehetőségei.
Ugyanis a külső szempontokat alapul vevő, előzetes rangsorolás helyett itt az alapvető kérdéssé az válna, hogy hogyan lehetne "általában" minél elérhetőbbé, rentábilisabbá tenni az újfajta (folyamatosan megújuló) technikát, szolgáltatást mindazoknak, akik azzal a saját elhatározásukból próbálkoznak.

A NI-TTM Csoport jövendő munkamódszerének kialakításában ez utóbbi változat adta előnyöket igyekszünk megteremteni, ezért kezdeményezzük a továbbiakban felvázolt vállalkozás megalapítását.

Miként lehet "segíteni" az "önállósághoz"? Az elhatározást, aktivitást, felelősséget pótolni vagy kiiktatni nem lehet. Lehet viszont könnyíteni, fejleszteni a körülményeket tájékoztatással, menedzselő szolgáltatásokkal és ügyeleti, inspekciós jellegű segítséggel (felkészülve a váratlan, katasztrófaszerű esetekre).

II.A megvalósítandó célkitűzés

A személyi számítógépek rentábilis működtetésének az elterjesztésével: a művelődést, a személyes-családi életvitelt, a háztartást segítő, humán jellegű (oktató, tájékoztató, ügyviteli stb, és rentábilis!) szolgáltatásoknak, valamint azok feltételeinek a kifejlesztése, elterjesztése, üzemeltetése.

III.A védjegyvállalkozás jellege

Ügynöki védjegy iroda. Hasonló példákra nehéz hivatkozni, nincsenek elterjedve.
A szemléletesség kedvéért szembeállítható a napjainkban divatos japán kereskedőházzal, amely (meglehetősen kiélezve a kérdést) a hatalmas tőkeerejénél fogva mintegy HELYETTESITI a finomabb piaci szerkezeteket. Leegyszerűsíti, monopolizálja a piacot, az eladó (termelő) és a vásárló (fogyasztó) közötti kapcsolatokat.
A védjegyvállalkozás ezzel szemben KATALIZÁLJA a piaci kapcsolatokat, folyamatokat, az ügyfelek egymás iránti bizalmát teheti könnyebbé, megalapozottabbá.
Az árut közvetlenül nem adja-veszi, hanem áttekinthetőbbé teszi az eladó és a vevő számára az egymáshoz való kapcsolatukat. Piaci kultúrát termel, sokféle áru adás-vételét teszi valószínűbbé kis tételekben is.
Csökkenti a közvetlenül az áru használata szempontjából diszfunkcionális, ám a csere, az eladás mozzanata miatt kényszerűen megfizetendő piaci költségeket (mind az eladó, mind a vevő számára).
A védjegyvállalkozás tehát az "egységes rendszerek" (hardver, szoftver stb) helyett a különböző rendszerek közötti mozgékonyságot, hangsúlyozza, ebben a, tőlünk függetlenül, folytonosan és egyre gyorsabban változó világban. Tehát a válogatást helyettesítő előírások helyett a személyi számítógépek piacán való eredményes, gyors tájékozódást segíti.

IV.A védjegyvállalkozás fő eszközei

1. Alapfeltétel a védjegyvállalkozás gazdálkodási önállósága.
2. Olyan szervezeti keretek kialakításának szorgalmazása, elsősorban a jelentkező művelődési házakban, amelyek hitelképessé, a rentábilis működtetésre szervezetileg alkalmassá teszik a számítógépek használóit.
3. Pályázatokkal olyan feltételek, szolgáltatások kifejlesztésének a szorgalmazása, amelyekkel elérhető a szolgáltatás bevételeiből finanszírozott számítógépüzemeltetés rentabilitása.
4. Védjegyhasználattal a kialakított szolgáltatások minőségének fenntartása, garantálása, megkönnyítve ezáltal a piacon való könnyebb tájékozódást az egyik részről, és a nagyobb piaci szolgáltatói fegyelmet a másik oldalról.
5. A szolgáltatások és termékek ismérveinek áttekinthető elemzési, tesztelési eljárásokba foglalása, közismertté tétele.
6. A piaci információk gyorsaságának, megbízhatóságának, közérthetőségének szorgalmazása (pl szakfolyóirat állandó rovata keretében).

V.Részletesebben a védjegyvállalkozás céljairól és eszközeiről

1.A rentabilitást megengedő és kimutató finanszírozási, működési feltételek

Széleskörű fejlesztési programokban egyre inkább célszerű a helyi erőforrások, kezdeményezések katalizálására törekedni. A rentabilitás az élet minden területén egyre időszerűbb gazdálkodási kérdés, hiszen csak a rentábilis működést lehet viszonylag szerényebb támogatással vagy banki hitelekkel sikeresen erősíteni.
A közművelődési célú személyi számítógép használat széleskörű elterjedésének, innovációs láncolatának a végső eleme a háztartás.
Jogos minimum-igényként felvethető a gyártótól a

forgalmazókon át a háztartásokig érő innovációs láncolat egészének a rentabilitása. Eszerint: egyik láncszem se lehessen rentábilis a többiek nélkül. Tehát a háztartás is járjon jól, az is legyen rentábilis.
A művelődési intézmények jelenlegi szabályozásában szinte lehetetlen beruházási hitelt felvenni. Enélkül viszont nagyon leszűkül azon intézmények köre, amelyek egyáltalán megvásárolhatják a rentábilis működtetésre alkalmas számítógépeket a saját elhatározásukból. Így viszont a művelődési ház sem lehet egykönnyen az innovációs láncolat rentábilis tagja (nemhogy a háztartás).
Fontos tehát megteremteni az eredményes gazdálkodás feltételeit mind a nagyobb művelődési házakban, mind a kisebb, egy-két személyes klubkönyvtárakban.

2.A rentábilitást valószínűsítő és igénylő közművelődési szolgáltatásterületek

Ebben az informatikai programban hosszabb távon célbavett háztartások informatizálása széleskörűen csak egy későbbi lépcsőfokban képzelhető el. Mai előfeltételként ehhez azonban alkalmas lehet a művelődési házak (és más, e feladatra vállalkozók) számítógépes szolgáltatásainak a kifejlődése mintegy az értelmes alkalmazási módok kísérleti terepeként, referenciájaként.
A művelődési háznak meg kell tanulnia kitermelni a szolgáltatások bevételeiből az általa használt hardver és szoftver árát. Ez szükséges a szolgáltatások érdemi kialakulásához, a referenciaszerep valóságához.
De kivel tudja megfizettetni? Található-e, gerjeszthető-e széleskörű fizetőképes kereslet a háztartásoknak szánt személyi számítógépes szolgáltatásokra?
Egy átlagos háztartás, vagy akár HATRÁNYOS HELYZETŐ (!) háztartás, ha olyan segítséget tud kapni a háztartásvitelhez, amelynek a segítségével ténylegesen megjavul a helyzete, akkor még a nehéz helyzetében is fizetőképes keresletet jelent!
Szokatlan, de lényeges feltevés, hogy, pl MENEDZSELÉSI JELLEGŰ vagy tájékoztató számítógépes szolgáltatásoknak az átlagjövedelmű, vagy éppen hátrányos helyzetű családok lehetnek a legbiztosabb fizetőképes kereslete. Azért, mert ők vannak a leginkább rászorulva erre a segítségre; szemben az elitjövedelműekkel, akik divatcikként is megengedhetik maguknak akár magának a számítógépnek a megvásárlását is. Anélkül, hogy ezen vásárlásuk megtérülésével, rentabilitásával szembe kellene nézniük.
Tehát éppen az átlagos vagy annál kisebb jövedelműek nehéz helyzete a garancia arra, hogy ismételten csakis olyan szolgáltatást fizetnek meg, tucnak megfizetni, amely tényleges változást, könnyebbséget hoz a helyzetükben.
Ilyen, szolgáltatásra alkalmas szoftvereket bizonyára nem könnyű előállítani ésjó színvonalon működtetni, karbantartani (főleg kezdetben). De reméljük, hogy nem is lehetetlen. Ezért merült fel a pályázatok gondolata a kifejlesztés segítésére.

3.Szolgáltatásfejlesztő pályázatok

A SIKERES SZOFTVEREK KÉSZÍTÉSÉT főleg a gyártók és a felhasználók közti kapcsolat javításával törekszünk hatékonyabbá és jelentősen olcsóbbá tenni.
Sokszor sikeresebb lehet az a szoftver, amit a felhasználókkal folytatott egyeztetések során készítenek el, esetleg olcsóbb is, mert a vevő szívesebben fordul az általa részben már ismert termékhez, hiszen a terjesztés, a vevőkeresés költségeit csökkenti az, ha megrendelésre, előjegyzésre lehet dolgozni. Ezen túl a termékek választékának bővülésére is számítani lehetne.
Nagyobb, munkaigényesebb bonyolult feladatok elvégzésére alkalmi munkacsoportokat összehozni is célja a pályázatnak. A szoftverpályázaton a sikeres pályázók részére díjak helyett elsősorban értékesítési csatornák felajánlása célszerű esetünkben.

AZ INFORMÁCIÓT SZOLGÁLTATÓK pályázata keretében a tájékoztató szolgáltatások információforrásaival keresik a kapcsolatot a pályázat kiírói.Az információk deklarált mérvű megbízhatósága elengedhetetlen a szolgáltatások működéséhez. Alapvető tehát az információt szolgáltatók érdekeltsége a tartós, szerződéses együttműködésben. Csak a tényleges érdekeltség alapján lehet hajlandó bármely információforrás a megbízhatóságot szankcionáló szerződést kötni.

A RENTÁBILIS ÜZEMELTETÉS pályázat keretében a pályázat kiírói a kényszerű szerepkiosztás helyett a vállalkozásra jelentkezőknek segítik a rentabilitás feltételeinek a tudatosítását, szükség esetén különféle tanácsadási, szervizelési lehetőséget biztosítanak a számukra. Fontos célkitűzés a jelentkezők részére bizonyos szervezési, menedzselési ismeretek átadása.
A pályázat keretében a pályázók egyúttal a siker és a kudarc referenciáit nyújtják egymásnak is, a kívülállóknak is.
Kiegészítheti az eddigieket a jövőben a számítógépgyártóknak, és -forgalmazóknak kiírt pályázat, bemutatkozási lehetőség.

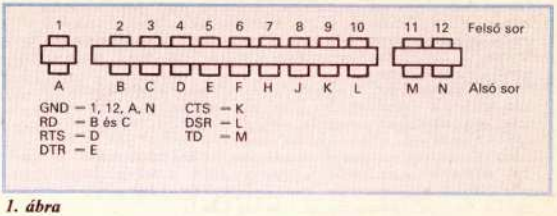
A PÁLYÁZAT TÁMOGATÁSÁRA bármilyen segítség elfogadható, ha a segítségnek nem feltétele a pályázat megváltoztatása (pl egyes termelők érdekében). A számítógépek, szoftverek stb vásárlására szánt esetleges juttatást vagy hitelkeretet csak olyan "külső" alapként lehet a pályázat vagy a védjegyvállalkozás részéről "kezelni", amely nem fut át a védjegyvállalkozás (pályázat) belső gazdálkodásán. A védjegyvállalkozás csak ezen pénzek elérhetőségét kötheti nyilvánosan kontrollált feltételekhez.

4.A szolgáltatások megbízhatóságát garantáló védjegy

A személyi számítógépek hardver- és szoftverpiacán már ma is sok termelő és gyártó működik. E piac további erős kibővülése esetén a gyártók és szolgáltatók száma várhatóan tovább nő. Egyre szélesebb körben használják a gépeket: a "titkokhoz" egyre kevésbé értő emberek.
Egyre fontosabb az áruk és a szolgáltatások könnyebb áttekinthetősége, megbízhatósága.
<Mint azt már az előbb is felhoztam: a pályázati keretben az újabb szolgáltatások kifejlesztése szorgalmazható, a védjegy üzemeltetésével pedig a már kialakított szolgáltatások színvonala, megbízhatósága erősíthető.>
A védjegy alkalmazásával meg lehet fogalmazni, és szankcionáltan garantálni lehet a védjegyhasználók körében olyan adás-vételi minőségi jellemzőket, amelyek egyébként az "egyszeri", vagy akár többszöri vásárlás esetén sem könnyen ellenőrizhetőek a laikus, "egyszeri" vevő részéről. Ha pontosan le van írva, hogy mit garantál a védjegy, és ezt a felek egyaránt el is fogadják, akkor legalább ebből a néhány szempontból kevésbé kell bizonytalankodnia a mégoly "egyszeri" vevőnek a vásárláskor.
A jogos bizalomhoz persze az is kell, hogy a védjeggyel ne lehessen tartósan visszaélni, hogy "éljen" a védjegy, működjön; azaz pl az előírások ellen vétőktől valóban megvonják a védjegy használatát. Fontos továbbá, hogy külön referencia-szolgáltatások tegyék megtanulhatóvá a követelmények teljesítését a védjegy újabb, leendő használói számára is.
Nemcsak a vevőt kell védeni az eladótól. Fordítva is felmerülhet a védjegyhasználat értelme, pl művelődési házak, mint vevők esetében (garantált fizetés stb). A számítógéppel végzett szolgáltatás is minősíthető védjeggyel (esetleg az előbbitől különbözővel) a közönség, a háztartások felé.Egy védjegy használata nem kötelező, hanem önkéntes. Tehát az előírásai is önként vállaltak, és így kérhetőek számon.
A védjegyhasználatot lehet szüneteltetni, és a feltételek újbóli megteremtése után meg lehet újítani.

Fáy Árpád
1985. október - 1986. január

VC20-ak egymás között



1. ábra

A VC20 csak alapgép, amely 3,5 kbájt szabad memóriájával nem sokat tud; de ha valaki egy kicsit is belemélyed lelkivilágába, mindjárt rájön, hogy mi mindenre képes ez a gép. A szokásos használata mellett többek között adatátvitelre is felhasználható ez a szerény képességű masina.

Természetesen az adatátvitel fogalma eddig is ismert dolog volt, de most kísérletet teszünk két VC20 összekapcsolására úgy, hogy azok egymás között adatforgalmat bonyolítsanak le.

Az igazat megvallva az ilyen párbeszédnek akkor van igazi jelentősége, ha például az egyik VC20 mér és a mérés eredményét feldolgozásra átadja a másiknak. Az adatátvitel megoldható postai vagy az ún. DATEX-hálózaton. Ha telefonvonalat akarunk felhasználni, modemet kell illesztenünk a gép és a telefonvonal közé, amely modulátor és demodulátor áramköröket tartalmaz. Nem túl nagy távolságok esetén gond nélkül megoldható az is, hogy párhuzamosan egy adat minden bitjét átvisszük, de ehhez kényelmetlenül sok vezeték kell. Ha soros adatátvitelt csinálunk RS—232C szabvány szerint, ez már csak 9 vezetéket igényel, meghatározott jelszintekkel. Az RS—232 arra szolgál, hogy a gépből jövő adatokat soros jelekké alakítsa át.

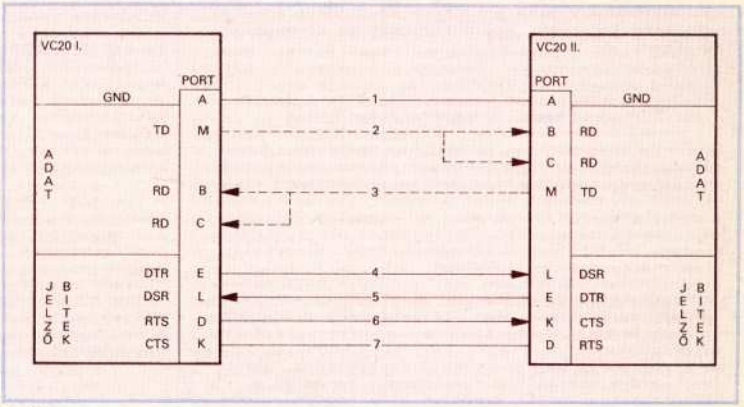
Az RS—232 interfész lényeges részeit a VC20-ba eredetileg beépítették, és minden olyan vezeték, amelyre szükségünk van, a USER PORT-on ki van vezetve. Itt a jelszint H = 5 V, L = 0 V.

Kisebb távolságok esetén az RS—232 elhagyható, és a két gép egymással is közvetlenül összeköthető. Az *1. ábrán* látható, hogy a USER PORT mely csatlakozásait használtuk fel. Az adatátvitelhez felhasznált jelek a következők:

- TD az elküldendő adatok kimenete;
- RD a fogadandó adatok bemenete;

A legegyszerűbb esetben ez a két vezeték is elég az adatátvitelhez, de az igazán biztonságos adatátvitelhez még jelzőbitekre is szükség van;

- DTR a logikai 1 szint azt jelenti, hogy a gép adatfogadásra kész;
- DSR a logikai 1 szint azt jelenti, hogy az ellenállomás adatfogadásra kész;
- RTS a logikai 1 szint azt jelenti, hogy az adatokat elküldik;
- CTS a logikai 1 szint azt jelenti, hogy az ellenállomás adatot akar küldeni (ez bemenet).



2. ábra

A két gépet a *2. ábrának* megfelelően kell összekötni. Kísérletképpen először csak a szaggatott vonallal jelzett vezetékekkel kössük össze a gépeket. Az adatátvitelt nagymértékben befolyásolja a vezeték induktivitása és kapacitása. Annál jobb, biztonságosabb az adatátvitel, minél alacsonyabb szinten tudjuk tartani a két fent említett értéket. Ajánlott árnyékolt vezetékek használata. Minél nagyobb adatátviteli sebességértéket (Baud/rate) választunk adott vezetékhossz mellett, annál bizonytalanabb az adatátvitel. A két gépet, ha lehet, ne forrasztásos eljárással kössük össze, hanem csatlakozókon keresztül, amelyek használata sokkal kényelmesebb.

Szöveg átviteléhez ez a program használható:

```
100 OPEN 2,2,0,
    CHR$(134)+CHR$(224)
110 GET#2,A$
190 REM OLVASÁS
200 GET#2,B$
210 PRINT B$;
220 IF B$<>"" THEN 200
290 REM TASZTATURA OLVASÁS
300 GET A$
310 IF A$ ="" THEN 200
320 PRINT A$;
330 PRINT#2,A$;
400 F=ST : REM STATUS SZÓ
410 IF F=0 OR F=8 THEN 200
490  REM HIBAJELZÉSEK
500 PRINT"hiba"
510 IF F AND 1 THEN
    PRINT"PARITÁS"
520 IF F AND 2 THEN
    PRINT"TÚLLÉPÉS"
530 IF F AND 4 THEN
    PRINT"FOGADÓ PUFFER TELE"
540 IF F AND 16 THEN
    PRINT"CTS HIBA"
560 IF F AND 64 THEN
    PRINT"DSR HIBA"
570 IF F AND 128 THEN
    PRINT"FOGADÓ MEGSZAKÍTÁS"
590 REM ADATÁTVITEL
600 IF PEEK(37151) AND 64=1
    THEN 600
610 CLOSE 2: END
```

Mind a két VC20-at fel kell tölteni ezzel a programmal. Az adatátvitel feltételét az OPEN paranccsal és a vezérlő-, valamint az utasításregiszter beállításával teremtjük meg.

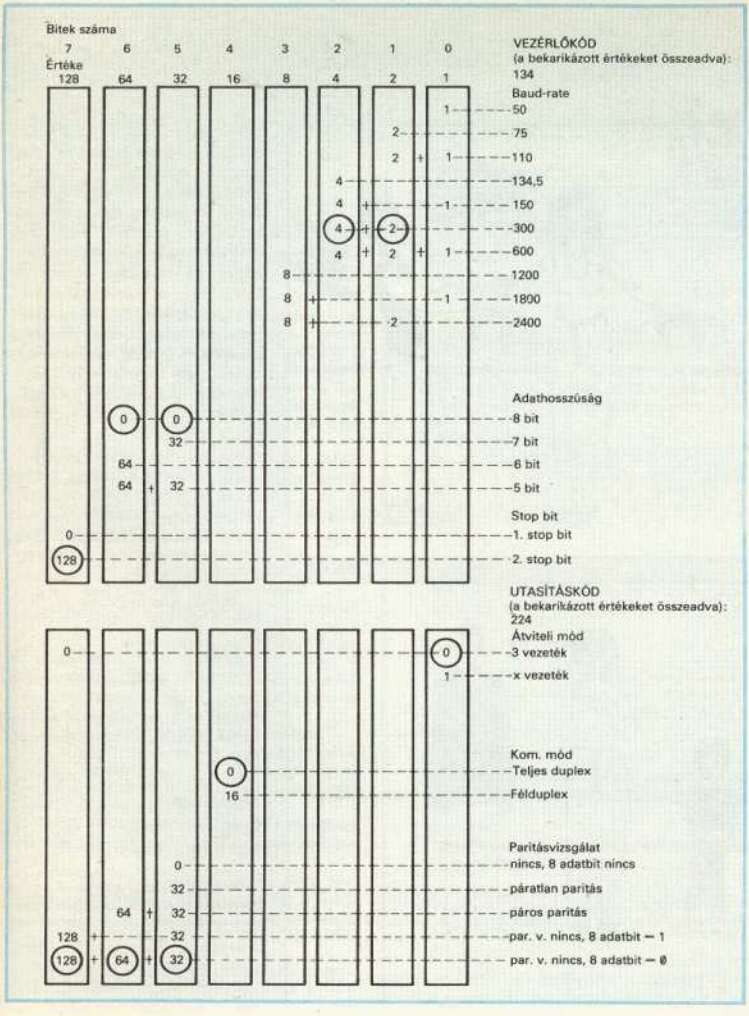
A vezérlőregiszter a következő feltételeket tárolja:

— Baud-rate érték, amely a VC20-nál 50 és 2400 között változhat. Start bittel és 1 vagy 2 stop bittel bájtonként 10, illetve 11 bitet viszünk át. 300-as sebességnél mintegy 27—30 karaktert, 2400 Baudnál 218—240 karaktert vihetünk át másodpercenként. Leggyakrabban használt a 300 Baudos érték.

— Adathosszúság: normál esetben 8 bit, de vannak olyan perifériák, amelyek 6 vagy még ennél is kevesebb bitet igényelnek.

— Stop bit: az adatátvitel 1 vagy 2 stop bittel fejeződik be.

Az utasításregiszter a következőket tartalmazhatja:



3. ábra

— „A kézfogás módja". Tulajdonképpen a két állomás közli egymással, hogy hány vezetéken történik a kapcsolattartás. Célszerű mind a két gépet azonosan beállítani!

— Átviteli mód: lehet teljes duplex vagy félduplex. A félduplex azt jelenti, hogy csak egyirányú adatforgalom van, tehát az egyik gép csak ad, a másik meg csak vesz.

— Paritásvizsgálat: az ellenállomás képes arra, hogy megvizsgálja, hogy az adatátvitel hibamentes volt-e. A paritásbit a 8. adatbit.

Az OPEN utasítás megnyitja az adatcsatornát, és beállítja az adatátvitel módját. Lehetőség van arra, hogy a készülékszámokkal más berendezés felé kormányozzuk az adatforgalmat.

A PRINT PEEK(659)-cel a vezérlőregiszter, a PRINT PEEK(660)-nal az utasításregiszter kérdezhető le.

Ha a *3. ábrát* figyelmesen áttanulmányozzuk, észrevesszük, hogy a vezérlő- és az utasításregiszternek egészen más értékeket adhatunk. A VC20 alapkiépítésében a kimeneti puffer a 7168—7423 és a bemeneti puffer a 7424—7679 címeken található meg. A bemeneti puffer kezdőcíme a PRINT PEEK (247)+PEEK(248)*256 és a kimeneti puffer a PRINT PEEK(249)+PEEK(250)*256 utasításokkal kérdezhető le.

A kommunikáció tulajdonképpen a ki- és bemeneti pufferek felhasználásával történik. A PRINT # 2 parancs segítségével a kimeneti pufferbe rakjuk az adatokat, majd megvárjuk, hogy az ellenállomás fogadókészséget jelentsen. Az adatátvitel az egyik kimeneti pufferből a másik bemeneti pufferbe történik. A bemeneti pufferben tárolt (megérkezett) adatokat a GET# 2 utasítással olvassuk ki. A 400-as sor a statusregisztert lekérdező rutint tartalmazza. A lekérdezés a BASIC ST szóval történik.

A 37151-es címen levő bájt segítségével megvizsgálható, hogy a kimeneti pufferben van-e még el nem küldött adat. A program mindaddig a 600-as sorban „kering", amíg ki nem ürül a kimeneti puffer, majd a CLOSE utasítás lezárja a csatornát.

Ha csak 3 vezetékkel, jelzőbitek nélkül kötjük össze a gépet, azok ugyan beszélgetnek egymással, de nem veszik figyelembe egymás vételkészségét, hanem véletlenszerűen történik az adatforgalom.

Természetesen nemcsak szöveges adatátvitelre használható fel a gép, hanem bármilyen más mérési és egyéb feladatokra is, csak akkor a programot is megfelelően módosítani kell. (A cikk a Funkschau cikkének felhasználásával készült.)

JÓKAI ANDRÁS

ADOK–VESZEK–CSERÉLEK

Ebben a rovatban rövid, szöveges, a mikroszámítógépekkel kapcsolatos hirdetéseket közlünk. A díjszabás: közületeknek gépelt soronként (60 karakter) 100,— Ft, magánszemélyeknek az első sor 50,— Ft, minden további sor 20,— Ft. Az NJSZT tagjainak az első három sor ingyenes. Hirdetéseiket a szerkesztőség címére várjuk.

■ **FÓLIAKAPCSOLÓK, TASZTATÚRÁK** gyártását vállaljuk. Grafikai kialakítás és az érintkezőpontok kapcsolása a megrendelő kívánsága szerint. Rövid szállítási határidő, kedvező ár! EL—CON GM, Budapest, Eperjes u. 90. IV. 12. 1204. Telefon: 287-109

■ **MSX RENDSZERŰ PHILIPS** személyi számítógép rekorderrel, vámkezelten eladó. Érdeklődni: Budapest XVIII., Széchenyi u. 124. 1184

■ **SCHNEIDER CPC 464** számítógép és Z80 nanocomputer eladó. Papp Antalné Telefon: 376-708

■ **COMMODORE 64-re** készült játékprogramokat cserélek. Cím: ifj. Gottmann István Bp., Mester u. 73. 1095

■ **COMMODORE 64-re** készült szoftver (program) -csere! Cím: Kótai Balázs, Sopron, Határőr u. 11. 9400

■ **KERESEM A KAPCSOLATOT** Timex-Sinclair 2068-as számítógép-tulajdonosokkal. Tapasztalatcsere! Cím: 7632 Pécs, Kassák u. 12. dr. Visegrády Lajos Telefon: 11-685

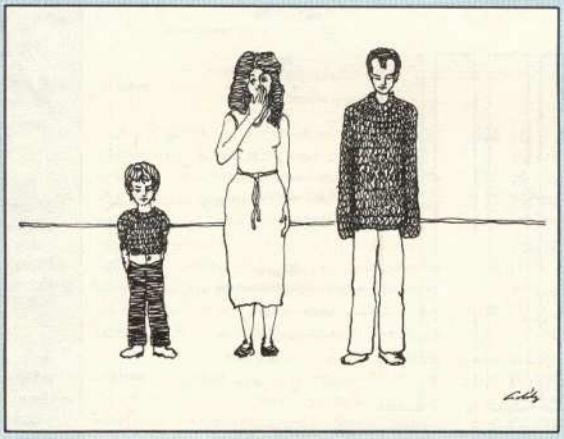
■ **ZX SPECTRUMHOZ** gépi kódú játékprogramot cserélek. Cím: Kovács Róbert, Tatabánya, Mártírok út 94. IX/3.

Hány sima, hány fordított?

Melyik gyakorló kötő nőt ne ért volna bosszúság azért, mert a divatlapból kinézett csodapulóverhez vásárolt pamutból éppen nem telt ki a második ujja? Melyikük nem bontott le hosszú estéken megkötött részeket, mert azok végül kicsik vagy ellenkezőleg, túl nagyok lettek? A divatlapok általában csak egyfajta fonalhoz, egy bizonyos tűmérethez adják meg a fonalszükségletet, és ezt mindössze egyetlen — általában az ideálisnak kikiáltott — méretre vonatkoztatják. Pedig például a kötés tömörségétől, tehát a szemek sűrűségétől is nagyban függ, hogy mennyi fonalat kell felhasználnunk. Biztos támpont hiányában sokszor rosszul kalkulálunk: vagy túl sok fonalat veszünk, vagy nem eleget.

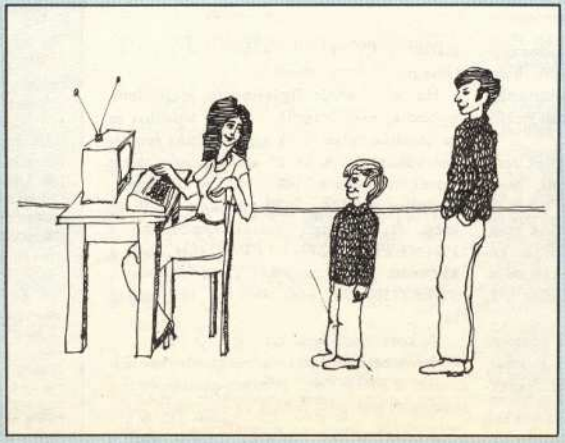
A kis Csizmadia marad a papa kaptafájánál

Csizmadia János közel 30 éve él az NSZK-ban. Néhány éve önálló programozóként dolgozik; professzionális személyi számítógépekre készít vállalati rendszereket. Tizenhét éves Tibor fia szenvedélyesen programoz. Egyik düsseldorfi magyar ismerősük, egy fonalbolt tulajdonosnője beszélgetés közben mesélt nekik a fent vázolt problémáról. Tibor azonnal felvetette, hogy számítógéppel meg lehetne ezt oldani. És kettőjüknek sikerült egy igazán „barátságos", pár hét alatt bevált rendszert kidolgozni, amely csalhatatlansága mellett még reklámot is jelent a hölgynek. Hiszen van-e hatékonyabb telekommunikációs eszköz, mint a női pletyka!? Mit is tud mesélni szomszédasszonyának, barátnőjének a számítógéppel kiszolgált vevő?



A kiválasztott fonalból a vevő saját kezűleg próbát köt, természetesen a megfelelő mintával és az általa választott kötőtűvel, majd leszámolják, hogy egy 10×10 cm-es darabon hány szemet kötött. És itt lép be az MZ700 japán, plotterrel felszerelt PC-re írt program.

A képernyőn feltűnik a bolt tulajdonosnőjének neve, címe, telefonja és egy invitáló üzenet: „Foglaljon helyet és feleljen a kérdésekre."



Rákérdez a vevő nevére, majd faggatja, mit szeretne: kardigánt, pulóvert vagy mellényt. Ennek eldöntése után a képernyő tartalma ismét választ vár a kézimunkázótól. Milyen fazont fog kötni: denevérujjút vagy hagyományos klasszikus fazont. Hosszú vagy rövid ujja legyen? Nyakkivágásának lehetőségei: V, kerek, csónak, zárt vagy garbó. Ezekből a képernyőre rajzolt alakzatok alapján választhatunk.

Ha mindezt közöltük a géppel, a képernyőn a kiválasztott fazon rajza, illetve a pulóver egyes darabjai jelennek meg, és a megrendelőnek be kell billentyűznie szükséges méreteit. A rajzok (méretezve) átkerülnek a plotter papírszalagjára is.

Miután ez is megtörtént, a képernyőn kívül a plotteren is kinyomtatja a program a rövidítéseket, amelyek majd a részletes instrukciókban szerepelnek.

Mibe kerül, hogyan fogy és szaporodik?

Először megkapjuk, hogy a bevitt feltételek mellett hány méter fonalra van szükségünk a kiválasztott darabhoz. Ebből ugye világos, hány gombolyagot kell megvennünk. És az igazi könnyebbség most következik: eleje, háta, ujja sorrendben, sorokra bontva megtudjuk, hány sima, hány fordított, mikor fogyasszunk és hol szaporítsunk.

Tessék? Hogy ezek a szakkifejezések nem érthetők a szenvedélyes számítástechnikus számára? Nos, kérdezzék meg a kedves feleséget vagy anyukát, aki torkig van vele, hogy férjét vagy csemetéjét szinte nem tudja elmozdítani a géptől. S ha a programot megismertetik vele, legközelebb ő is szívesen számoltatja ki az ajándékpulóver pamutszükségletét. Vagy félnek, hogy ennek nyomán esetleg konkurens „felhasználóként" lép majd fel a géptől eddig távol maradó családtag?

ti

Még egyszer vissza az ékezetekhez

Akárcsak az 1985/2. számukban, az Ember—gép kapcsolat rovatban megjelent hozzászólás, a mellékelt íráskép is Sinclair Spectrum számítógéppel, szintén a TASWORD 2 egy „magyarított" változatával, a nyomtatás pedig Centronics GLP mátrixprinteren készült. Az ennél az írásnál alkalmazott megoldásnak egy hátránya és egy előnye van a hivatkozott cikkben ismertetetthez képest.

A hátrány: nem felel meg teljesen az MSZ 9212 szabványnak, nem nyomtatható hosszú ő és ű betű, nincs nagy hosszú í, ó és ú. A nagy á betű egy „ortopéd" svéd betű. Ez a hátrány azonban minden olyan nyomtatónál kiküszöbölhető, amely engedelmeskedik a CHR 8 (kurzor balra) utasításnak. A GLP sajnos nem „veszi be" ezt az utasítást.

Az előny: a szövegszerkesztés és nyomtatás a nyomtató átalakítása nélkül, kizárólag a TASWORD 2 program átalakításával és kibővítésével történt, méghozzá úgy, hogy a TASWORD eredeti funkciói — egy kivétellel — továbbra is teljesülnek, és a szövegfájl sem rövidül meg.

Ehhez két feladatot kellett megoldani:

1. Gondoskodni kellett arról, hogy bizonyos „feláldozott" ASCII karaktereknek megfelelő billentyű leütésére a képernyőn megjelenjenek az „ékes" betűk.

2. El kellett **„intézni"**, hogy a nyomtató **ezeket a megváltozott** karaktereket vigye papírra.

Az első feladat semmi gondot nem okozott, mert csak át kellett alakítani a TASWORD 2 saját apróbetűs (64 karakter/sor) karakterkészletét, a 61184 — 62079 memóriatartományban.

A második feladat megoldásának magyarázatához már bele kell tekintenünk a TASWORD 2 lelkébe. A programot BASIC bevezető rész vezérli (RAMTOP=32000). Az 54782 memóriacímig a szövegfájl (320 64 karakteres sor, összesen 20480 ASCII karakterkód), a szerkesztési terület és a két képernyőt betöltő mankó foglalja el. Az 54784-es címtől a fizikai RAMTOP-ig terjed a gépi kódú programrész.

A TASWORD a 32-től 127-ig terjedő CHR kódú karakterek kinyomtatására képes. A 128—143 CHR kódokat (a Spectrum blokkgrafikáit) a nyomtató vezérlésére használja. A 144—164 kódtartományt (Spectrum felhasználói grafikák) nem ismeri fel, míg a magasabb kódtartomány egyes elemei a szövegszerkesztő funkciókat vezérlik.

Mivel a GLP-n az ékezetes karakterek a TASWORD által fel nem ismert vagy ki nem nyomtatott tartományba esnek, a legcélszerűbb megoldásnak az látszott, ha a RAMTOP-ot alacsonyabbra véve, egy külön kódtranszformáló — nyomtató, gépi kódú programot iktatok be. Az ehhez a szöveghez használt változatnál ez a programrész mindössze 460 bájt, és alkalmas mind a nyomtatásra kerülő karakterek transzformációjára, mind a nyomtatásvezérlő „blokkgrafikák" felismerésére.

```
Ez most normál.

1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.
Ez NLQ

1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.

Ez NLQ nagyitott

1234567890!AéöüáéöÜ_qwertyuiop<>;
UIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.

 Ez ismét normál
Ez aláhúzott
1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.
  ez nagyitott


1234567890!AéöüáéöÜ_qwertyuiop<>;
UIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.

ez kettős leütés

1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.


ez fett

1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.

Ez sűrített
1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.

Ez  normál
1234567890!AéöüáéöÜ_qwertyuiop<>;"QWERTYUIOP[]Σasdfghjkl^ASDFGHJK
Li-+=~σμ§τzxcvbnmZXCVBNM:ò?/ú,.
és lapdobás


alsó index és felső index

P  [ x < q ]  = ∑  [ (ln /q/ - μ) / σ  ]
```

Technikai okokból a nyomtatásvezérlő kódoknál nem alkalmazható a TASWORD 2 opciós megoldása, hanem összesen 16 előre meghatározott (set vagy reset) utasítás lehetőségét tartalmazza a program. Ez azonban gyakorlatilag ki is meríti a legtöbb nyomtatónak a szövegszerkesztésben ténylegesen hasznosítható lehetőségeit. A jelen változatban például az NLQ, azaz megközelítő levélminőség, alsó vagy felső index, aláhúzás, nagyított, kettős leütésű, fett és sűrített betűk, valamint a lapdobás vezérelhető a TASWORD-ból.

A GLP karakterkészlete lehetővé tette néhány görög betű beiktatását is. A fixen beépített nyomtatásvezérlő grafikákkal például az ábra utolsó sorában látható formulák is kinyomtathatók.

Jelenleg olyan univerzális TASWORD kiegészítőn dolgozom, ami az angol ábécé 52 betűje és az arab számok, az írásjelek és előjelek mellett a legtöbb printer karakterkészletének bármely max. 18 elemű részhalmazának kinyomtatására alkalmazható.

Ezt a megoldást előnyösebbnek, olcsóbbnak és univerzálisabbnak tekintem a nyomtatók átalakításánál.

DR. FRIGYES ERVIN

■

A szerkesztő megjegyzése: Az ilyen — a valóságot nem szépítő, használható megoldást adó — cikkek nincsenek ellentétben az 1986/2. számban leírtakkal (ti. azzal, hogy sem a C16, C64, VC20, sem a Sinclair gépeken nem valósítható meg szöveges üzemmódban a *teljes* magyar ábécé), ezért szívesen közöljük azokat.

KEDVES OLVASÓINK!

A μM-hoz érkezett levelek nagyobbik részére a szerkesztők közvetlenül válaszolnak, ebbe a rovatba már csak azok kerülnek, amelyek közérdekű javaslatot tartalmaznak vagy ilyen kérdést tesznek fel.

Sokan kérdezték, hogy lehet a TV-BASIC tanfolyam ismétlése után vizsgázni. Jelentkezési lapot kell beküldeni, amelyet a μM-ban, az Ötletben és a Rádió- és Televízióújságban fogunk közölni. Úgy, mint a tavalyi vizsgánál, mindenki értesítést kap, hogy hol kell a vizsgára jelentkeznie. Elegendő jelentkező esetén a vizsgát megrendezzük valamennyi megyeszékhelyen, Budapesten pedig a legtöbb kerületben. A tavalyi vizsga anyagát elküldtük a μkluboknak; akit érdekel, elkérheti.

Dr. Simonyi Endre lakástelefonja megváltozott. Új száma: 556-245.

Dr. Fiók János, Budapest,

Kazinczy u. 2. 1191

Ezúton szeretnék jelentkezni a μ'86 nevű mikroszámítógépes találkozóra, amelyen szívesen bemutatnám néhány C64-re írt programomat.

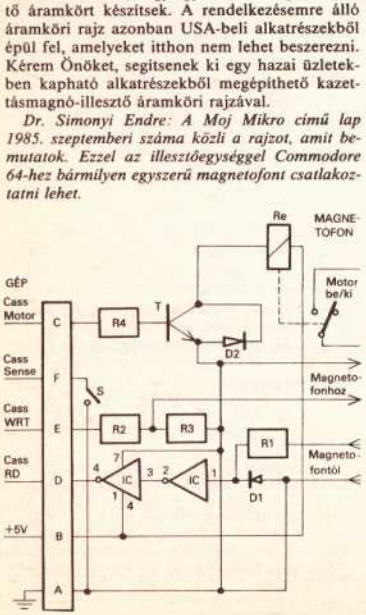
Reméljük, sokan tervezik programjaik, gépépítő ötleteik bemutatását a μ'86-on. Mindenkit nagy szeretettel várunk.

Fábián Kornél, Debrecen,

Eötvös u. 65. 4029

Tavaly kaptam egy C64-est, de sajnos datasett nélkül. Elektroműszerész tanuló vagyok, ezért vállalkozom arra, hogy egy kazettásmagnó-illesztő áramkört készítsek. A rendelkezésemre álló áramköri rajz azonban USA-beli alkatrészekből épül fel, amelyeket itthon nem lehet beszerezni. Kérem Önöket, segítsenek ki egy hazai üzletekben kapható alkatrészekből megépíthető kazettásmagnó-illesztő áramköri rajzával.

Dr. Simonyi Endre: A Moj Mikro című lap 1985. szeptemberi száma közli a rajzot, amit bemutatok. Ezzel az illesztőegységgel Commodore 64-hez bármilyen egyszerű magnetofont csatlakoztatni lehet.



Az ábra jelölései:

S — kapcsoló; IC — 74LS14; D1 és D2—1N4148 vagy 1N914; Re — 5 V-os relé; R 1 — 100 ohm; R 2 — 220 ohm; R 3 — 22 kohm; R 4 — 1 kohm; T — BC 140, B 141.

Zbyšek Bahenský U plynárny 12,

Praha 4 145 00 P. O. Box 30

A prágai Villamosipari Főiskola 21 éves hallgatója vagyok. Van egy ZX—Spectrum gépem és nagyon szívesen programozok, sok programom van otthon. Most külföldi programcsere-lehetőséget keresek. Nemrég voltam Budapesten, és ott számítógépklubokat kerestem, de akikkel találkoztam, nemigen tudtak ilyen klubról, sem kapcsolati lehetőségekről. Kérem, küldjék el nekem a Sinclar-használók klubjának vagy egy Sinclair-használónak a címét, akivel Spectrumra programokat, ötleteket vagy megoldásokat cserélhetek. Sajnos nem beszélek magyarul; németül vagy barátom segítségével angolul tudok írni.

Természetesen levélben válaszoltam. Miután már több hasonló tartalmú levelet kaptam külföldről és belföldről egyaránt, felsorolom az NJSZT HCC szekcióvezetők nevét. A leveleket az NJSZT titkárság címére küldjék, ahonnan a címzettek azonnal megkapják.

A HCC elnöke:	*Dr. Simonyi Endre*
A HCC titkára:	*Diebel Dietrich*
Homelab—Aircomp szekció:	*Barabási Rezső*
Sinclair szekció:	*Török Zoltán*
Apple szekció:	*Diebel Dietrich*
TRS 80—HT szekció:	*Szupkai István*
M 6800 szekció:	*Dr. Simonyi Endre*

Akiket a Spectrum programcsere érdekel, azoknak azt tanácsolom, vegyék fel a kapcsolatot Zbyšek Bahenskýval, aki biztosan szívesen küld programokat.

Sulák Zoltán, Székesfehérvár,

Nyitrai út 55. 8000

A lap korábbi számaiban több olvasó panaszolta, hogy kevés a megjelenő Commodore program. Ezután e programok száma megnövekedett, természetszerűleg kiszorítva a Spectrum és egyéb programokat. (Nem vagyok pártatlan, én is Spectrum-tulajdonos vagyok.) Azóta némi ingadozás vehető észre, de a mérleg nyelve mindig a Commodore felé mutat. Spectrumhoz, kevés kivétellel, főleg játéklistákat közöltek, holott hasznosabbak lennének más jellegű (oktató-, statisztikai vagy ha játék-, akkor hosszabb lélegzetű gépi kódú) programok.

Középiskolába járok, idén érettségizem. Iskolánkban két HT és két Primo üzemel. Az előbbiekkel semmi gondunk, ám a Primókhoz nincs semmiféle irodalmunk, kivéve a kezelési utasítást, ami elég szegényes. A gépi kódú programozáshoz pedig igen hasznos lenne a gép memóriatérképe, továbbá egyéb „beavatkozásokhoz" hardverkönyv. Segítségüket, tanácsukat kérem: hol lehet ezeket beszerezni, megvásárolni.

Valóban egyre több programot kapunk C64-re, aminek az oka egyszerű: a jelenleg kapható olcsó és az iskolákban is használt gépek közül ennek a legjobb a grafikája, egyelőre erre a gépre programoznak legszívesebben. Ha jó, eredeti és érdekes programot kapunk bármelyik másik (Primo, Sinclair, HT) gépre, gyorsabban közöljük, mint a C64 programokat, hiszen azokból mindig van elegendő. Levelének Primóra vonatkozó részére nem tudok válaszolni, elküldtem a COSY leányvállalat illetékeseinek.

Kovács László, Dunaújváros,

Rádióamatőr Computer Club, pf. 24. 2408

Előző levelemben írtam az RCC megalakulásáról és arról, hogy még rengeteg megoldatlan problémánk van a klub működésével kapcsolatban. Szeretnénk, ha a klubon keresztül a dunaújvárosi számítógép-tulajdonosok egymásra találnának, és így megalapoznánk egy korlátlan programcsere-lehetőséget. Én elsősorban a dunaújvárosi Commodore 64 tulajdonosokra gondolok, mert a helyi Munkás Művelődési Központban van egy Spectrum klub. Szeretnénk beindítani a C64 oktatást a klubtagok részére. Tervezzük, hogy bérmunkában szoftverkészítést is vállalunk. Ehhez szükség volna egy helyiségre, amelyben heti 2-3 alkalommal tarthatnánk a foglalkozást. Amíg nem sikerül a klub részére a megfelelő hardverkapacitást beszerezni, addig csak a géppel rendelkezőkre számítunk, programcseréket, tapasztalatcseréket bonyolítunk le. Akit Dunaújvárosban érdekel a klub, annak jelentkezését várjuk a postafiók címére. Szeretnénk majd a különböző szekcióinknak (programcsere, oktatás, levelezés, szerviz-ügyintézés stb.) vezetőket választani. Közlöm még, hogy klubunknak van egy felesleges Commodore fényceruzája, amelyet szívesen elcserélnénk más mikroklubokkal.

Levele azért is tanulságos, mert azt hiszem, hogy az országban működő μklubok hasonló módon fejleszthetnék tevékenységüket. A tanfolyamokat biztosan szívesen fogadnák a környékbeli üzemek is, de a szoftverfejlesztést sem tartom rossznak. Szeretnénk többet hallani a hazai μklubokról. Tevékenységük leírását szívesen közöljük. Kérdéseire levélben válaszoltam.

Király László, 5230 Sömmerden,

Fichte Str. 15. Pf. 05—03 NDK

Köszönetet mondok a Magyar Postának, hogy lehetővé teszi a külföldön élő magyarok részére folyóiratok, magazinok olvasását. Eddig csak néha-néha sikerült a μM-t megszerezni. Az 1985/5. számtól folyamatosan kapom.

Magazinuk munkatársakat keres. Én szívesen lennék a munkatársuk. Foglalkozásom műszerész, a Robotron vállalatnál dolgozom DX—RX

6311—6320 mátrixnyomtatón. Nyelvtudásom révén fordíthatnék Önöknek (mindenkinek) cikkeket, érdekesebb beszámolókat, bár én inkább javítással, kiegészítő berendezésekkel foglalkozom. A munkahelyemen Z80, otthon C64 áll rendelkezésemre. Az utóbbi időben sokat foglalkoztam 3.D grafikával. Az itt használt segédprogrammal szerzett tapasztalataimat is szívesen közzétenném.

A válaszom ismét többeknek is szól. Minden cikket szívesen fogadunk. A lapban megjelent írások jó részét nem a szerkesztőség munkatársai készítik. De előre senkinek, még saját munkatársainknak sem adunk garanciát, hogy amit írtak, az megjelenik. Mint minden lapnál, nálunk is a felelős szerkesztő dönt, hogy végül mi kerül a lapba, és mi nem. A döntéshez pedig el kell olvasni a cikket. A nem közölt írásokat nem küldjük vissza. Nagyon sok levelet is kapunk, ezeknek egy részét közöljük, sokra válaszolunk is, de nem mindegyikre, hiszen nem győznénk idővel. Olvasóink tanácsait, javaslatait elolvassuk, nagyon sokat már meg is valósítottunk. Ezt bizonyára észrevették.

Szilágyi Róbert, Andornaktálya,

Petőfi út 3. 3399

Köszönöm az előző levelemre adott gyors választ és a benne közölt információkat. Az MK 27-es kiválóan együttműködik a Spectrummal. Az Önök magazinja a legjobb lehetőség a számítógéppel való ismerkedés, tanulás és játék programozásának elsajátítására. Eddig megjelent számaik közül a legelső és az 1985/2. hiányzik. Legyenek szívesek tájékoztatni, hogyan szerezhetném meg ezeket. Ha esetleg elküldenék, nagyon megköszönném. Még egy tanácsot kérek. A µM 1985/5. számából szereztem tudomást egy könyvről, amit szeretnék megvásárolni: Ligeti—Szervánszky: A ZX—Spectrum programozása. Kérem, segítsenek a megszerzésében, mert itt Egerben és környékén nem ismerik ezt a kiadványt.

Néhány régi számból az NJSZT-ben még van néhány példány, sajnos az első még „aranyért" sem kapható. A következő számokból tudunk küldeni: 1984. évi 1., 2., 3., 4. szám, 1985. évi 3., 5. és 6. szám. A ZX—Spectrum programozása című könyv a SZÁMALK kiadványa. Forduljon közvetlenül a SZÁMALK Könyvesbolthoz (Bp. XI., Szakasits Árpád út 68.).

Gál István, Debrecen,

Tőzsér u. 3. 4031

Kérem, hogy küldjék el nekem az 1985/5. számban megjelent Torpedó nevű játék listáját. A lapban ugyanis — printerrel való nyomtatás miatt — a lista egyes részei nem látszanak vagy nem olvashatók. Hasonló hibák az előző számokban is vannak. Javaslom, hogy a listákat úgy írják, mint a két reflexjátékot és az Űrutazás nevű játékot.

Többen panaszkodtak már a programlistákra. Egy jó barátom, maga és családja a lap egyik legkritikusabb olvasója, minden szám megjelenése után közli, hogy „a listák még mindig pocsékok." Kaptunk tőle egy javaslatot, remélem, sikerül megvalósítani, akkor ezeket a hibákat felszámoljuk.

Marton Gábor 8. oszt. tanuló, Boglárlelle,

Sallai u. 31. 8638

Köszönöm, hogy elküldték a µM 1984/4. számát, mivel a sorozatomból csak ez hiányzott. Még egy kéréssel fordulok Önökhöz. Írjanak valamit lapjukban a PC 128-ról. Úgy gondolom, ez a gép a Commodore cég jól sikerült gépe, és engem is nagyon érdekel, gondolom, nem vagyok egyedül.

Általában akkor írunk egy-egy számítógépről, ha abból elég sok üzemel az országban, és egy olyan szerzőt találunk, aki nem a prospektusokból írja meg a cikkét, hanem az adott gépen megfelelő tapasztalatot szerzett. Valószínűleg a Commodore 128 is sorra kerül.

Nagy Zoltán, Zalaegerszeg,

Ősz u. 6. 8900

12 éves vagyok, van egy C16-os gépünk. Kevés programunk van, és sajnos én sem tudok rá még programot írni. Ezért kérem, hogy közöljenek játékprogramot a Magazinban, amit mi is rendszeresen járatunk.

A C16-os gépek nemrégen kerültek be az országba, ezért nem közöltünk rá programokat. De ami késik, nem múlik.

Dr. Rózsa Pál, egyetemi tanár, tanszékvezető

BME Villamosmérnöki kar, Matematika tanszék

Kérem a t. Szerkesztőséget, hogy az újságban rendszeresen megjelenő „Az olvasó írja" rovatban tévesen közölt információt figyelembe venni szíveskedjék, s a továbbiakban a listájukról töröljék. A HCC Commodore szekció: BME H ép. III. em. Számítógépszoba páros héten csütörtökön 17 órától meghirdetett programja valótlan, a tanszéken sem ebben, sem más időpontban foglalkoztatás nincs, és előreláthatóan nem is lesz.

Az információ a HCC-től származik, a téves közlésért és az esetleges kellemetlenségekért szíves elnézést kér

KOVÁCS GYŐZŐ

ORTEKON

Tervezési
és
Fejlesztési Gmk

ORTEKON GMK

Telefon:
154-250, 887-861,
Bányai

1036 Budapest, Árpád fejedelem útja 69.

Vállaljuk:

— műszerek, villamos és elektronikus ipari termékek konstrukciós és technológiai tervezését, komplex felszerszámozását a sikeres „0" sorozatig;
— elektronikus (digitális) vezérlőegységek, szabályozókészülékek és rendszerek tervezését és gyártását;
— a gyártás-előkészítés és -elszámolás irányítási és adatfeldolgozási feladatainak megvalósítását több munkahelyes számítógéprendszer alkalmazásával;
— CNC vezérlés komplett felújítását.

A megfelelő referenciákról személyesen adunk tájékoztatást.

Rendszerjellemzők alakulása

Több rendszernek a modelljét adtuk meg kapcsolási rajzával, operátorainak kapcsolatábrájával. Az operátorcsapatok mindegyikében van késleltető operátor. A rendszerek bemeneti rendszerjellemzőinek alakulási folyamatát leíró függvények formulája az ábrákon a bemeneti jellemzőket képviselő körök mellett van.

A feladat a többi rendszerjellemző alakulási folyamatára vonatkozóan az, hogy ki kell válogatni azokat a rendszerjellemzőket, amelyek a megadott bemeneti folyamatok mellett úgy változnak, hogy

- korlátosak maradnak;
- abszolút értékük tetszőlegesen nagy értéket (is) felvesz;
- tartanak valamilyen véges értékhez (más szóval *konvergálnak valamilyen véges értékhez*, vagyis *„állandósulnak"*).

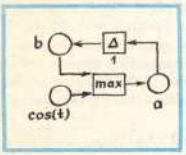
Az összes rendszer működtetését a 0 időponttól indítjuk. Az összes késleltető a kezdeti időpontban állandó függvényt tartalmaz, amelynek értéke zérus. (Azaz a késleltetők 0-val vannak feltöltve.)

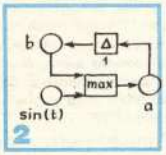
A *több-bemenetű* maximumképző és minimumképző kimeneti értéke minden időpontban a bemeneti értékeinek ugyanebben az időpontban felvett maximumával, illetve minimumával egyenlő.

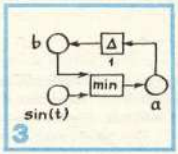
Az *egybemenetű* maximumképző és minimumképző kimenetén minden időpontban az a legnagyobb, illetve legkisebb bemeneti folyamatérték szerepel, amit a kezdő időponttól a szóbanforgó időpontig tartó zárt időintervallumon a bemeneti folyamat felvett. (Az egybemenetű maximumképző kimeneti folyamata tehát nem csökkenő, az egybemenetű minimumképzőé pedig nem növekvő.)

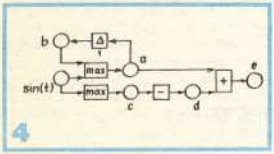
Az összes feladatban t a mindenkori időpontot mutatja és 0-tól + ∞-ig változik.

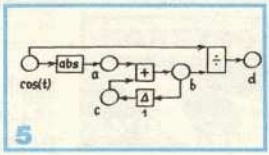
TAKÁCSY ILDIKÓ

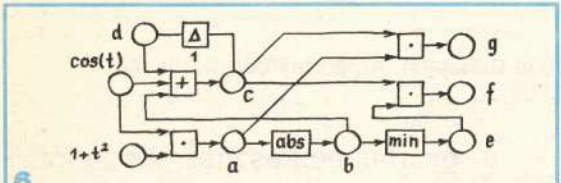


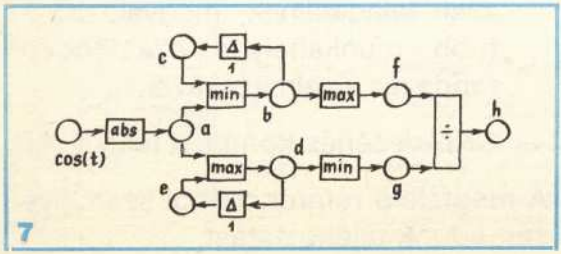


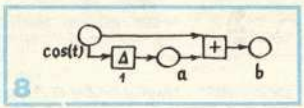


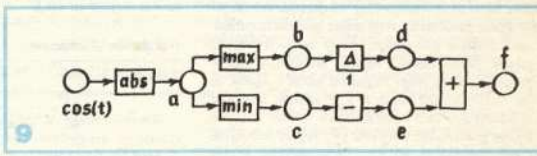


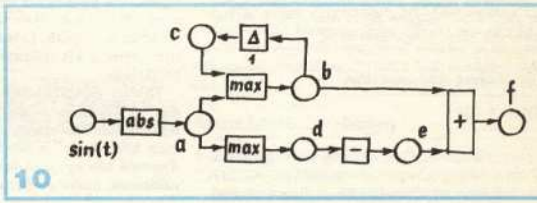


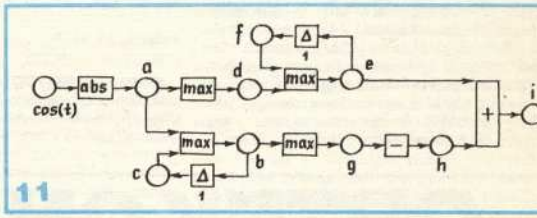


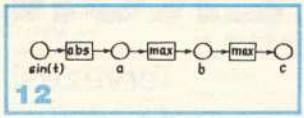


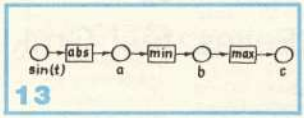


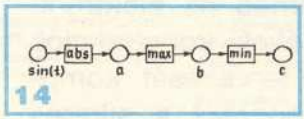


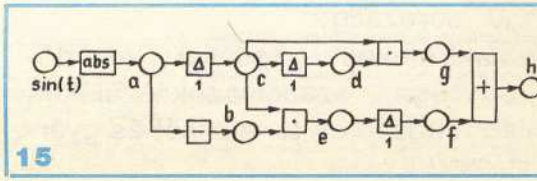












MEGOLDÁS

Kerítéses mintaként: 1. a, b, 2. a, b, 3. a, b, 4. a, b, c, d, e, 5. a, d, 7. a, b, c, d, e, f, g, h, 8. a, b, 9. a, b, c, d, e, f, h, i, 12. a, b, c, 13. a, b, c, 14. a, b, c, 15. a, b, c, d, e, f, g, h.

Tárt valamilyen véges értékhez (ha állandó, vagy állandóvá válik; azt is ide számítottunk): a→1, 2. a→1, 3. a→1, b→1, 4. a→1, b→1, c→1, d→1, e→0, 5. d→0, 6. e→0, f→0, 7. b→0, c→0, d→1, e→1, f→0, g→cos(1), b→0, 9. b→1, c→0, f→0, 10. b→1, c→1, d→1, e→1, f→0, 11. b→1, c→1, d→1, e→1, f→1, g→1, b→1, i→0, 12. b→1, c→1, 13. b→0, c→0, 14. b→1, c→0, 15. h→0.

Abszolút értékek között tetszőlegesen nagy érték (is) előfordul: 5. b, c, 6. a, b, c, d, g.

A számítógépek különleges képességei

Sokak vásárlási döntését a készülék, illetve program játékerején kívül az is befolyásolja, hogy tud-e a számítógép valami mást is, mint sakkozni velünk vagy egy másik számítógéppel szemben. Sőt vannak, akik számára az ilyen különleges képességek fontosabbak, mint maga a játék.

Elemzőképesség

Első helyen kell említeni a számítógép elemzőképességét. Közismert, hogy minden sakkszámítógépnek, -programnak „fokozatai" vannak. Sok cég azzal hirdeti készülékét, hogy milyen sok — mondjuk 8, 9 — fokozata van. Tudni kell, hogy a fokozat nem más, mint a készülék számára biztosított, lépésenkénti átlagos gondolkodási idő. Nem túl nagy a jelentősége annak, hogy a készüléknek hány fokozata van, de megkívánjuk, hogy legyen közöttük két-három gyors, 10, 20, 30 másodperces gondolkodási idővel; okvetlenül képes legyen játszani a versenyeken előírt, lépésenként átlag 3-4 perc gondolkodás mellett (2, illetve 2,5 óra alatt 40 lépés); és feltétlenül legyen „végtelen" fokozata, amikor nem lép, csak elemez, akár több órán át, és végül, amikor megszakítjuk, megmutatja, hogy hány lépés mélységig történt elemzés után mely lépést ítéli a legjobbnak. A közbenső fokozatok (például 1, 2 vagy 5, 10 perc átlagban egy lépésre) szinte mellékesek, a gyártótól függ, hogy melyeket választja. Ezeket a sakkozó ritkán használja. Felismerte ezt *David Levy*, az ismert *Mark V.* és *VI.* típusú, kitűnő sakkszámítógép tervezője és programozója. Készülékének nincsenek fokozatai, de be lehet állítani rajta, hogy átlagban mennyi ideig gondolkozhat egy lépésen.

A programozók legfőbb törekvése, hogy optimális játékerőt érjenek el versenyfokozaton. De egyáltalán nem közömbös, hogy milyen erősen játszanak gyors játszmában, hiszen otthon szívesebben játszunk könnyű partikat gyorsan, mint versenytempó mellett. És ennél is fontosabb, hogy milyen a program, a készülék elemzőképessége. Eljut-e mondjuk néhány órán belül addig, hogy kellő mélységig elemezve az állást, megtalálja a legjobb lépést.

Sokan szeretnek készülékükkel teszteket végezni, felállítani olyan hadállásokat, amelyekben valami szép kombináció, váratlan fordulat elvezet a győzelemhez. Vajon a számítógép megtalálja-e a nyerő folytatást, és ha igen, mennyi idő alatt? A legérdekesebb a teszt akkor, ha előbb 3 percet adunk a készüléknek, hogy lássuk, megtenné-e a jó lépést versenyjátszmában; ha ez nem sikerül, adjunk több időt, végül állítsuk végtelenre, hogy egyáltalán bekerül-e az optimális lépés vizsgálódásának körébe, és ha igen, mennyi idő elteltével.

Sokat elárul egy elemző teszt a számítógép játékerejéről is. Fontos a levelezési sakkozók számára, akik közül többen veszik igénybe számítógépek segítségét. Benne hagyják az állást a készülékben akár egy-két napig is: vajon mire jutnak?

Vannak, akik ezt nem tartják etikusnak. Ugyan miért? Levelező sakkozók akár nagymester tanácsát is meghallgathatják. Aztán pedig: aki feltétel nélkül a számítógépre hagyatkozik, nem vizsgálja lépéseit kritikus szemmel, könnyen ráfázhat.

Tudunk olyan levelező sakkozóról, akinek öt különféle típusú számítógépe van, és mindegyikbe betáplálja a soron következő játszmahadállást, majd azt a lépést küldi meg ellenfelének, amelyre a többség „szavaz". Kivéve, ha ő maga mégsem azt a lépést tartja a legjobbnak.

Feladványfejtés

Csaknem minden sakkszámítógépnek és -programnak van speciális feladványfejtő fokozata. Ennek különlegessége abban áll, hogy algoritmusa eltér a normál sakkozó fokozatokétól. Értékelő funkciója leegyszerűsített, semmiféle anyagi vagy pozíciós előnyt nem vizsgál, kizárólag arra törekszik, hogy a legrövidebb időn belül mattot adjon.

Nem érdektelen a készülékeket problémafejtő képességükön keresztül is összehasonlítani: melyikük a gyorsabb. Egy jó mikroszámítógép a kétlépéses feladványokat néhány másodperc, a hármasokat pár perc, a négyeseket is legfeljebb egy-két óra alatt megfejti. Egy hosszabb lépésszámú feladvány megoldása sok órát, egy-két napot is igénybe vehet. A cégek sokszor így hirdetnek: a számítógép 7 lépésen belül minden feladványt megfejt. Azt azonban nem teszik hozzá, mennyi idő alatt.

Van a feladványfejtéssel kapcsolatban egy nagyobb probléma. A legtöbb készülék, ha megtalált egy megoldást, amely megfelel az előírt követelménynek (például: matt két lépésben), nem vizsgálódik tovább, hanem leáll. Nem keresi az esetleges további megoldásokat, pedig előfordul, hogy egy feladvány hibás, többféleképpen is megoldható. És a számítógép feladványfejtő képességének legfőbb gyakorlati haszna éppen az lenne, hogy feltárja a művek esetleges hibáit. Erre a piacon eddig megjelent számítógépek közül kb. 2 évvel ezelőtt csupán a már említett *Mark V.* és *VI.*, valamint az *Elite* volt képes. Legújabban mind több olyan készülék jelent meg a piacon, amely szintén hibátlanul fejt.

A „korrekt" fejtés nemcsak a feladványok specialistái számára lényeges, hanem mindenkinek, mert könnyen előfordulhat, hogy a készülék egy semmitmondó mellékmegfejtést talál meg, amely annak tulajdonképpeni tartalmából semmit sem árul el. Az egymással való összehasonlításra sem alkalmasak a csupán egy megfejtésig gondolkodó készülékek, mert a vizsgált lépések sorrendje véletlenszerű, az algoritmus sajátosságaitól függ, és elvileg előfordulhat, hogy a sok ezer lehetséges lépés közül véletlenül az elsők között, vagy éppen elemzése végén bukkan rá a számítógép a megfejtésre. Az összehasonlítás akkor reális, ha a program minden lehetséges variációt kielemez.

A sakkfeladványok alkotói, valamint a rovatvezetők és a versenybírók számára a számítógépek feladványfejtő képessége — túlzás nélkül állíthatjuk — korszakos jelentőségű. Elősegíti, hogy a szerzők hibátlan feladványokat alkossanak, hogy a rovatvezetők az ellenőrzés nélkül beküldött művek hibáit megállapíthassák, és ne közöljenek mellékmegfejtéses feladványokat, a versenybírók pedig ne adjanak díjat vagy egyéb kitüntetést hibás műnek. Sokszor derül ki mostanában számítógép révén egy-egy évtizedekkel ezelőtt megjelent, és eddig korrektnek tartott mű hibája. Nemrég is előfordult a Magyar Sakk Szövetség egy elbírált feladványversenyén, hogy a versenybíró a sakktábla mellett, gép nélkül ellenőrzött 157 pályaműből 17-et talált kitüntetésre érdemesnek, és miután nyomtatásban leközölték őket, a számítógép 10-ben (!) hibát fedezett fel, és a hibásakat utólag diszkvalifikálni kellett.

Ehhez azonban nemcsak az említett sakkszámítógépek, hanem személyi számítógépekre írt, speciális feladványfejtő programok is rendelkezésre állnak. Ezek gyorsabban fejtenek, és nagy előnyük, hogy a legkülönbözőbb „műfajú" fel-

adványhoz értenek, a mattfeladványokon kívül a szabadmatthoz, önmatthoz stb. is; hadd ne soroljuk itt. Igen értékes válfaja ez a sakkprogramoknak. (A világon az egyik legelső, hibátlanul fejtő programot — amint már korábban ismertettük — *Szálka Imre*, a Számítástechnikai Koordinációs Intézet tudományos munkatársa készítette.)

Szimultán játék

Be lehet programozni egy sakkszámítógépet arra is, hogy szimultán játszmákat váltson. Erre csak az olyan hardver alkalmas, amelynek kijelzője vagy képernyője van; ezen sorra, egymás után megjelenhetnek az egyes játszmák lépései és hadállása. Az ilyen játék igen érdekes, mindig nagy a közönségsikere. Néhány szimultán bemutatót Magyarországon is rendeztünk.

Tulajdonképpen semmi különös nincs a szimultán játékra képes programban, memória kérdése az egész. Ugyanis több játszma lépéseit kell egyszerre tárolni és egymás után „elővenni". Mégis, eddig kizárólag a *Mark V.* és *VI.* típusú készüléket programozták be erre. 12 játszmát tud egyidejűleg váltani. A legcélszerűbb a készüléket legfeljebb 20 másodperces lépésenkénti gondolkodási időre beállítani, így már elfogadhatóan játszik, és egy 12 játszmás szimultán — a lépések betáplálására és a résztvevőknek történő bemondására vagy kiírására szükséges időt is figyelembe véve — néhány óra alatt lebonyolítható.

A számítógépek szimultán játéka egy tényezőben lényegesen különbözik az emberétől. Míg a sakkmester figyelmét — a leggyakorlottabbét is — sok játszma egyidejű vezetése feltétlenül szétszórja, a számítógépnek tökéletesen mindegy, hogy egy vagy több játszmát vált-e egyidejűleg. Csak az a fontos, hogy elég erős legyen a gyors játékban is.

Ez a látványos „szolgáltatás" azért csupán egyetlen számítógép sajátja, mert ennek ábrakijelzője van, amelyen az állás váltogatható; a korszerű, sakktáblában rejlő számítógépeken csak egy játszma lépéseit lehet megtenni, jelezni. Reméljük, a hardverek tervezői megtalálják a módot e probléma áthidalására.

DR. LINDNER LÁSZLÓ

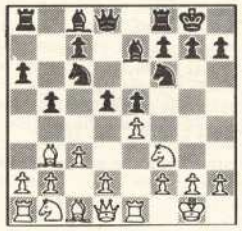
BITEK ÉS FIGURÁK

Lépésgenerálás II.

Előző számunkban megismerkedtünk a táblaindexes ábrázolási módszerrel végrehajtott lépésgenerálás két fajtájával, a halmazmező és a táblavezérlési módszerrel. Most harmadikként a lépéslista folyamatos megújításáról lesz szó.

A lépéslista folyamatos megújítása lényegében szintén a halmazmező vagy a táblavezérlésű lépésgenerálást alkalmazza, azzal a különbséggel, hogy egy-egy újabb hadállásnál nem számítja ki az összes lehetséges lépést, hanem az előző hadállás lépéslistáját újítja meg. Ez azért előnyösebb, mert egy lépés megtétele az előző állásban listázott legális lépéseknek körülbelül az egyharmadát változtatja meg, kétharmadára nézve közömbös marad. Ez a módszer a másik kettőnél gyorsabban generálja a lépéseket, viszont nagyobb tárolót igényel. Vizsgáljuk meg az *ábrán* látható hadállást. Nézzük meg, hogy sötét legutóbbi lépésére — amely D7—D5 volt — mely figurák lépéslistáját kell megújítani:

b) a C8 futóét és a D8 vezérét, amelyeknek vonala megnyílt; az F6 huszárét, amely a gyalog kiindulási mezejére léphet;



Úgy hisszük, egyáltalán nem volt nehéz megállapítani, hogy mely figurák lépéslistájának megújítása volt szükséges. Az éppen megtett lépés által mozgásukban befolyásolt figurák megtalálására az alábbi szabályt állíthatjuk fel.

Újra kell listázni a következő figurák lehetséges lépéseit:

- azét, amely lépett,
- amelyet leütöttek (ennek lépései törlendők),

azokét, amelyek

- a lépett figura kiindulási mezejére tudnak lépni, illetve a lépett figura által megnyitott vonalon vagy átlón a kiindulási mezőn át tudnak haladni,
- a lépett figura érkezési mezejét támadják, illetve a tett lépés következtében elzárt vonalon vagy átlón az érkezési mezőn immár nem tudnak áthaladni,

végül azét a gyalogét, amely

- a kiindulási mező mögött egy (vagy alapállású gyalog esetén két) mezővel áll, és amely
- az érkezési mező előtt egy (vagy alapállású gyalog esetén két) mezővel áll.

A programnak csak a fent felsorolt figurák lépéslistáit kell ellenőrizni (természetesen nem biztos, hogy valamennyi típusból van a táblán), és előállítani az utolsó lépés hatására megváltozott hatókörű figurák megújított „félig legális" lépéslistáját.

Hasonlítsuk össze a megismert három módszert. Míg az első kettőnél minden féllépésben az esedékes színű valamennyi figura lépéslistájának előállítására szükség van, addig a lépéslista folyamatos megújítása esetén minden féllépésben mindkét színű, de csak a fentiekben felsorolt fi-

```
;***** SAKKPROGRAM: LEPERV szubrutin *****
;* keszitette:KOVACS ATTILA, LOVASS LASZLO *
;*****************************************
;
;A LEPERV szubrutin ellenorzi a jatekos lepesenek
;ervenyességet a szabályos lepesek listaja alapjan
;
LEPERV LD   A,(PSZIN)       ;  veszi a jatekos szinet
       XOR  #80             ;  (=pszin (7,BIT))
       LD   (SZIN),A        ;
       LD   HL,(LLISM)      ;  elmenti a lepeslista mutatojat
       PUSH HL              ;
       LD   HL,LLINX-2      ;  bealiitja a kovetkezo
       LD   (LLISX),HL      ;  lepeslista indeket
       LD   HL,LLST+KVLST   ;  veszi a kovetkezo mutatot
       LD   (LLKV),HL       ;
       CALL LEPGEN          ;  listat keszit a jatekos
       LD   IX,LLST+KVLST   ;  ervenyes lepeseirol
       JR   LP2             ;
LP1    LD   C,(IX+LLIM)     ;  veszi a kovetkezo lepest
                            ;  a listan
       LD   B,(IX+LLIM+1)   ;
       XOR  A               ;  ha a listanak vege van
       CP   B               ;
       JR   Z,LP5           ;  ugras LP5-re
       PUSH BC              ;
       POP  IX              ;
LP2    LD   A,(LEPK)        ;  a 'honnan pozicio' egyezik
       CP   (IX+HNANL)      ;
       JR   NZ,LP1          ;  ha nem,veszi a kovetkezo
                            ;  lepest
       LD   A,(LEPK+1)      ;  a 'hova pozicio' egyezik?
       CP   (IX+HVAL)       ;
       JR   NZ,LP1          ;  ha nem, akkor jon a kovet-
                            ;  kezo lepes
LP3    LD   (LLISM),IX      ;  vegrehajtja a lepest, es
       CALL LVEGR           ;  ha utana sakkban van, akkor
       CALL SAKKE           ;
       AND  A               ;
       JR   NZ,LP4          ;  ugrik az LP4 cimre
       JR   LP6             ;  ervenyes lepes eseten visz-
                            ;  szater
LP4    CALL LVISSZ          ;  visszaveszi a lepest
LP5    LD   A,1             ;  jelzi a szabalytalan lepest
       POP  HL              ;
       LD   (LLISM),HL      ;
       RET                  ;  visszater
LP6    POP  HL              ;
       RET                  ;
```

```
;*************************************************
;A SZINR rutin megkeresi a tablatombben az adott szinu
;figurakat
;*************************************************
SZINR   LD    A,21          ;  a tablatomb 1. eleme
SZK1    LD    (LO),A        ;  tablaindexet hoz letre
        LD    IX,(LO)
        LD    A,(IX+TABL)   ;  veszi a mezo tartalmat
        AND   A
        JR    Z,SZK2        ;  ha a mezo ures,ugrik SZK2-re
        CP    -1
        JR    Z,SZK2        ;  ha hatarolomezo,ugrik SZK2-re
        LD    (FO),A
        LD    HL,SZIN
        XOR   (HL)
        BIT   7,A           ;  megegyezo figura szin eseten
                            ;  visszater
        RET   Z
SZK2    _D    A,(LO)
        INC   A             ;  veszi a kovetkezo mezot
        CP    TVEG          ;  ha a tombnek nincs meg vege,
        JP    NZ,SZK1       ;  akkor SZK1-re ugrik
        LD    A,1
        OR    A             ;  Z jelzobit 0
        RET
;*************************************************
;A SAKKE rutin ellenorzi,hogy az adott szinu kiraly
;sakkban van-e
;*************************************************
SAKKE   CALL  SZINR
SE1     AND   7
        CP    KIR           ;  az adott szinu figura kiraly?
        JR    Z,SE2         ;  ha igen, ugrik SE2-re
        CALL  SZK2          ;  folytatja a keresest
        JR    SE1
SE2     LD    (L1),IX       ;  indexes
        LD    (TO),A        ;  figura tipus tarolas
        CALL  TAMD          ;  megkeresi a figura
                            ;  tamadoja(i)t
        RET
;*************************************************
;A LEPGEN rutin osszegyujti adott szinu figurak
;osszes lepeseit
;*************************************************
LEPGEN  LD    DE,(LLKV)
        CALL  SAKKE         ;  ellenorzi,hogy a kiraly
        LD    (SAKKB),A     ;  sakkban van-e
        LD    HL,(LLISX)    ;  beallitja a lepeslistat a
        INC   HL            ;  kovetkezo fellepesre
        INC   HL
        LD    (HL),E
        INC   HL
        LD    (HL),D
        INC   HL
        LD    (LLISX),HL
        LD    (LILK),HL
        CALL  SZINR
LG1     CALL  Z,FIGM        ;  osszegyujti az adott figura
        CALL  SZK2          ;  osszes lepeset
        JR    Z,LG1         ;  addig keres ujabb figurat,
        RET                 ;  amig vegig nem ment a tab-
                            ;  la tombjen
;*************************************************
;Egyeb szubrutinok:
;-LVEGR ... megteszi a lepest a tabla tombjen
;-LVISSZ... visszavszi a lepest a tombon
;-TAMD .... megkeresi az adott figura tamadojat
;-FIGM .... osszegyujti az adott figura osszes lepeset
;*************************************************
```

gurák „félig legális" lépéseit kell megújítani.

Hogyan tárolja a számítógép ezeket a lépéslistákat?

A táblaindexes, illetve a táblavezérlésű módszernél a program a belső táblát letapogatja az A1 mezőtől a H8 mezőig valamilyen sorrendben (például soronként), és az így kiszámított „félig legális" lépések halmazát a tároló valamely részében egymás után, *a letapogatás sorrendjében* elraktározza. A lépéslista folyamatos megújításánál viszont célszerű a „félig legális" lépések halmazát *figuránként* elraktározni a tárolóban. Így minden egyes figurához egy-egy rekord rendelhető, és a program csak az esedékes rekordot — amelyet a fenti szabály nyolc pontja szerint talál meg — írja át az aktuális lépés megtétele után. Egy ilyen rekordban a program más adatokat is tárolhat, ami szintén a lépésgenerálás utóbbi módszere mellett szól, de erről egy későbbi számban lesz szó.

A lépésgenerálás halmazmező módszerére írtuk Z80 Assembly nyelven a programot, amelyet egyébként a Kempelen-sakkprogram Sinclair Spectrum változatában használtunk volna fel. A programlistából egy-két triviális szubrutint kihagytunk, de ezek funkcióját és működési módját a programban jelezzük.

KOVÁCS P. ATTILA

PROPOS–16 operációs rendszer, alapismeretek

(Budapest, 1985. SZKI, 132 oldal)

Egy három kötetben készülő kézikönyv első kötete ez; elsősorban a számítástechnikában még járatlan kezdőknek szól.

Kezdők bevezetése a számítástechnikába nem egyszerű, félvállról vehető feladat. A szakmai pontosság igényei mellett azonos súllyal jönnek számításba a didaktikai követelmények: a logikai egymásra építés, a fogalmak érthető magyarázata, a szövegezés világossága, félreérthetetlensége és sok minden egyéb. Nos, úgy tűnik, hogy kisebb hibáktól eltekintve ez a könyv minden szempontból jól sikerült. Mindenekelőtt meg kell jegyezni, hogy a kezdők tanításában oly lényeges motiváló hatást igen példamutató módon éri el, elsősorban a kép- és rajzanyagával. A kezdő részére kissé száraz témát annyira érdekessé, kedvessé teszik a rajzok, hogy a könyvet azok is átlapozzák, akik egyébként szakmailag az anyag színvonala felett állnak. Didaktikai szempontokból igen sikeres a tipográfia is, mely nemcsak a lényeges tudnivalók kiemelését segíti, de egyfajta rendszerszemléletet is sugall.

Általában a tartalmi rész is jónak mondható, bár egy-két dolgot jobban is meg lehetett volna oldani. Így például az utasítás és a parancs, mint a számítógép központi egysége részére előírt tevékenység, nincs világosan megmagyarázva, a kezdő olvasó részére összemosódik ez a két élesen elválasztandó alapfogalom. Helytelen az a beállítás is, mintha a parancsot csakis operációs rendszer megléte esetében lehetne végrehajtani.

Összemosódik a „lemezkönyvtár" (directory) és a „gyökérkönyvtár" terminusok használata a megfelelő értelmezések hiányában. A könyv használja a „fordítóprogram" fogalmat, ami elkerülhetetlenül szükséges is. Az azonban fogyatékosság, hogy nem magyarázza meg a fordítás mibenlétét, sőt gépi kódról, kompilációról sem esik szó, hogy az interaktív üzemmódot ne is említsük. E tekintetben egy fogalmi, értelmezési szakadékot látok, ami a kezdő részére az önálló könyvhasználatot ezen a ponton szinte lehetetlenné teszi.

Nem lényegbevágó, de a logikai építkezés szempontjából jogos igény, hogy ha egyszer közöl a könyv egy ASCII kódtáblázatot (ami nem lett volna feltétlenül szükséges), akkor adjon magyarázatot a grafikus jelek kódjaihoz.

Nagy előnye a kötetnek, hogy a végén értelmező szótárt nyújt olvasójának. Erre még a szakmailag haladó olvasók kézikönyvében is szükség van, kezdőkében pedig nélkülözhetetlen. A szótárban az értelmezések túlnyomó többsége világos, egyszerű és pontos fogalmazású. Egy-két címszónál azonban javításra szorul: például a memória nemcsak központi, hanem háttértár is lehet. A rekord és a fájl kapcsolatának magyarázata sem eléggé világos a kezdő olvasó részére. Hiányolom az értelmező szótárból az „adat", „azonosító", „információ", „katalógus", „mező", „virtuális tár" és néhány további címszó, értelmezés szerepeltetését.

Összegezve az elmondottakat, a könyv a számítástechnikai oktatás jelenlegi hazai felkészültségét tekintve jónak mondható. Bizonyos szempontokból kiváló, egyes kérdésekben egy második kiadásban javításra szorul. Az olvasók kevés külső segítséggel önállóan is használhatják.

ADA-WINTER PÉTER

Felföldi József–Sz. Lukács Sándor: HT–1080Z iskolaszámítógép fix tárolója

(Budapest, 1985. Ipari Informatikai Központ, 315 oldal. Ára: 220,— Ft.)

A könyv hiánypótló, mert a BASIC interpreter és a monitorprogram működéséről részletes magyar nyelvű leírás eddig még nem jelent meg.

A kötet két fő részből áll. Az első a HT–1080Z iskolaszámítógép 12 kbájtos ROM-ját írja le. Az egyes rutinok kezdőcíme és tartalma külön is ki van emelve. Ez megkönnyíti a felhasználást. A fontosabb segédrutinok címe után a működés lényegét összefoglaló leírás található. A második rész az EPROM-mal foglalkozik. A függelék táblázatai: az interpreter által kezelhető kulcsszavak; a fontosabb rutinok címe; a rendszerváltozók; a program és a változók tárolásának módja; a matematikai függvények kiszámításának módja; a blokkdiagramok és a ROM eltérések. Ez utóbbi ismerteti, hogy a kezdetben gyártott gépek ROM-ja hol és miben tér el a könyvben ismertetett változattól.

A szerzők feltételezik a Z80 mikroprocesszor utasításainak és az assembly programozás jelöléseinek ismeretét. Figyelembe veszik, hogy a felhasználók nagy része nem szakember, de nem is kezdő, így az egyes megjegyzések megfogalmazásánál a lehető legtömörebb leírásra törekednek. A bonyolultabb, nehezen követhető algoritmusokat folyamatábra formájában is ábrázolják.

A könyv segítséget nyújt a sokszor hiányos értelmezéssel közölt programlisták visszafejtéséhez, és lehetőséget kínál programozási fogások elsajátításához. Olyan információk gyűjteménye, amelyek ismeretében a középiskolások is nekivághatnak a nagy szakmai felkészültséget kívánó feladatok megoldásának.

SIEGLER GÁBOR

Margaréta-kerekes nyomtató

1985 őszén mutatták be a szocialista országok első margarétakerekes mikronyomtatóját EVMOLPIA 100 néven. A Plovdivi Írógépgyár fejlesztette ki, 1984 tavaszától gyártott margarétakerekes írógépének, az EVMOLPIA 40-nek IZOT 6912 néven forgalmazott, mikroszámítógépekhez Centronics illesztővel ellátott változatából. Az EVMOLPIA 100 alapvetően nem is különbözik az elektronikus írógéptől, csupán a billentyűzettől fosztották meg. Nyomtatási sebessége 18 jel másodpercenként, súlya 15 kg. A szocialista országok közül az NDK-beli robotron cég margarétakerekes mikronyomtatójának megjelenése várható még a közeljövőben.

Lakáscsere

Az Egyesült Államokban számítógéppel sokoldalúan támogatják a lakáscserét. Az már klasszikus megoldásnak számít, hogy az érdeklődők kívánságaik ismertetése után kapnak egy címlistát az aktuális kínálatról. Ezután a kliens nyakába veszi a várost, és végigjárja a címeket. Sok helyre azonban már be sem megy, mert nem tetszik az utca, a környék vagy maga a ház.

Ezt a problémát kiküszöbölendő a hirdetéseket bonyolító cég videofilmre veszi a város minden utcájának mindkét oldalát. A filmet aztán a gigabájtokban mérhető tárolókapacitással rendelkező lézerlemezre másolja úgy, hogy közben számítógéppel értelmezhető pontokat raknak le. A klienseknek nyújtott szolgáltatás így már nem a címlista átadásával zárul, hanem kívánságra az érdeklődő a képernyőn megtekintheti az adott utcáról felvett videofilmet is, sőt megállítva a szóban forgó ház felbukkanásakor a képet, akár el is bámészkodhat a leendő lakhelye előtt.

16 bit – hosszú távon

Az előrejelzések szerint nem valószínű, hogy a jelenleg legjobban terjedő 16 bites mikroszámítógépeket gyorsan ki fogják szorítani a valóban 32 bites mikroszámítógépek.

A lépésváltás elég lassú. Például még 1984-ben is a 8 bites gépek képviselték a kereskedelmi forgalom 47 százalékát. A becslések szerint 1990-ig összesen 16 millió 16 bites számítógépet adnak el. 1990-ben az értékesített 5 millió mikroszámítógépből várhatóan csak mintegy félmillió lesz 32 bites. Ezt a lassú terjedési ütemet a szakemberek azzal magyarázzák, hogy a felhasználók a 16 bites gépek teljesítményét elegendőnek tartják feladataik elvégzéséhez.

A 16 bites mikroszámítógépek piacán pénzértékben kifejezve az IBM szerezte meg a forgalom csaknem 50 százalékát. Az IBM nyugat-európai országokba irányuló szállításainak értéke évente átlagosan 30 százalékkal növekszik.

Szakértői rendszerek

Mivel hiány van a magas fokon képzett szakemberekben, a mesterséges intelligencia kutatói olyan módszereket és eszközöket keresnek, amelyek a szűk keresztmetszeteket áthidalják. Ezek közül gyakorlati alkalmazásuk szempontjából a legjelentősebbek a szakértői vagy más néven tudásalapú, ismeretalapú rendszerek.

Ezek két vonatkozásban különböznek a hagyományos számítógép-alkalmazásoktól. Míg a klaszszikus gépeket az operációs rendszer és az alkalmazói program működteti, a szakértői rendszer egy ismeretbázisból és egy, a következtetéseket végrehajtó alrendszerből áll. Az előbbi a szóban forgó problémával kapcsolatos tényeket, szabályokat tartalmazza, az utóbbi értékeli a szabályokat.

A szakértői rendszereket elsősorban orvosdiagnosztikai célra használják, de vannak már más, például gépkocsi-diagnosztikai, tanácsadó könyvtárosi és tervezéstámogatási rendszerek is.

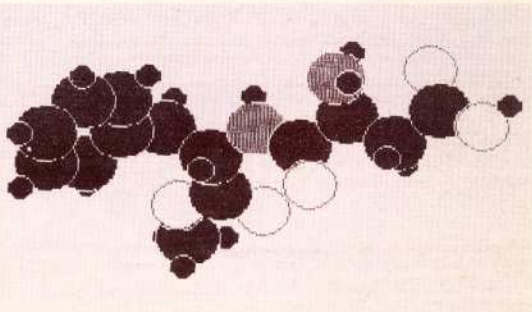
Integrált szoftver

Az irodai számítógép-alkalmazás egyik legtipikusabb feladata, amikor az adatbázisban tárolt adatok alapján a szövegfeldolgozó program segítségével jelentést kell készíteni, és abban táblázatosan, sőt grafikusan közölni bizonyos részeket. A szoftverfejlesztő vállalatok ilyen feldolgozási célokra készítették az integrált programcsomagokat, amelyek közül a legnépszerűbb és kereskedelmi szempontból is a legeredményesebb a LOTUS 1-2-3. Ma már egyre többféle ilyen programtermék kapható.

LABSWARE

A COMPUDRUG Kisszövetkezet olyan, a laboratóriumi munka minden területére kiterjedő, egységes felépítésű programgyűjteményt készített személyi számítógépre, amely jelentős mértékben megkönnyíti a vegyészek, gyógyszerészek, mezőgazdászok, orvosok, a környezetvédelemmel foglalkozók és a vezetők napi munkáját, és segítséget nyújt az oktatásban is.

A fejlesztésnél alapvető szempont volt, hogy a számítástechnikában járatlanok is könnyen alkalmazhassák a programokat. Az alapváltozatokat az egyes szakterületek elismert szakemberei készítették, ami garantálja a magas szakmai színvonalat. A programok segítik a hatóanyag-tervezést, a farmakokinetikai, a biometriai és a klinikofarmakológiai tevékenységet. A programok mind CP/M, mind MS—DOS operációs rendszer vezérlése alatt futnak.



A MOLMOVIE nevű, a molekulák felépítését, mozgását oktató, szinkezelési lehetőségeket is tartalmazó program nyomtatón készült képernyőmásolata

Óriás teljesítmény piciben

A DEC bemutatta a 32 bites VAX család legújabb tagját. A MicroVAX II környezeti igényeit tekintve valódi mikroszámítógép, azaz légkondicionálást nem igényel. Teljesítménye viszont csaknem eléri a VAX 11/780-ét.

Széles felhasználói területre szánják, például valós idejű feldolgozásra, irodaautomatizálásra, számítógéppel támogatott tervezésre. A sikeres piacra hozásának záloga a teljes programkompatibilitása a nagyobb VAX rendszerekkel, azaz bármely, a VMS-gépekre, a VAX 11/730, 750 vagy 780-ra kidolgozott program változtatás nélkül fut rajta.

A valós idejű feldolgozásokhoz a modulárisan kiépíthető VAXELN operációs rendszert ajánlják. A MicroVMS operációs rendszer teljes kompatibilis a VAX/VMS-sel, attól csupán moduláris felépítésében különbözik. A MicroVAX ULTRIX — 32 m a UNIX operációs rendszeren alapul (4.2 verzió), elsősorban a sokfelhasználós alkalmazásokhoz ajánlják. Operatív társa 4 Mbájtig építhető ki, és már megoldották a 30—70 Mbájtos lemezek illesztését. A kiterjesztett Q-busz alkalmazása leegyszerűsíti a gép konfigurálását és a számítógépnek hálózatba integrálását.

Személyiszámítógép-javítás, karbantartás közületeknek, magánszemélyeknek.
Egyedi megrendelés alapján kiegészítő berendezések gyártása.
Pl. Sinclair fényceruza, Joystick Interface, oktatási intézménynek kabinet kialakítása.

Pásztor Ferenc személyiszámítógép-javító és -karbantartó kisiparos. Szolnok, Mátyás király u. 2. V/3.

INFORMÁCIÓTECHNIKAI VÁLLALAT

SZÁMÍTÓGÉPKÖZPONTOK

figyelmébe ajánljuk
a svájci kooperációban készült:

Form —Tronic—ITV márkajelzésű
leporellofeldolgozó berendezéseket,

amelyeket a legmodernebb integrált építőelemek alkalmazásával
hosszú élettartalmú üzemeltetést garantálnak.

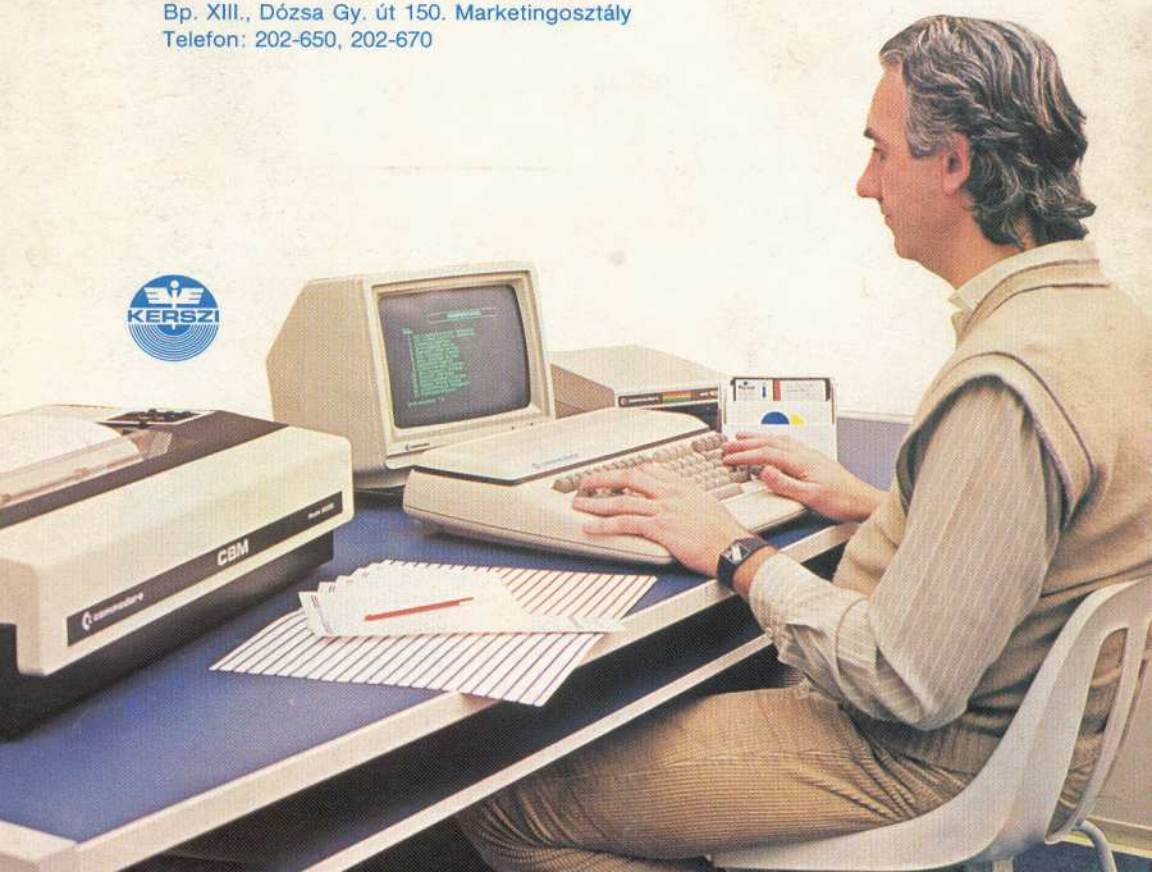
Alkalmazásának előnyei:
GYORSASÁG, PONTOSSÁG, MEGBÍZHATÓSÁG,
nagy teljesítmény, egyszerű kezelés

műszaki jellemzői:
— elektronikus vezérlés,
— univerzális beállítás,
— optimális technológia

Vásárlási igényekre
részletes tájékoztatással a KERESKEDELMI FŐOSZTÁLY
az érdeklődők rendelkezésére áll:

Budapest III., Kerék u. 4. Telefon: 803-294

A KERESKEDELMI SZERVEZÉSI INTÉZET új bemutatótermében a következő programok és programcsomagok tekinthetők meg, melyek COMMODORE 610-es személyi számítógépre készültek:

1. 15 fajta kartonrendszer vagy egyéb nyilvántartás számítógépes megoldása. Ára: 22 000 Ft
2. Állóeszköznyilvántartási rendszer. Ára: 40 000 Ft
3. Jövedelemérdekeltségű boltok meghatározott mutatóinak nyilvántartása és elemzése. Ára: 20 000 Ft
4. Az 1986. évi vállalati jövedelemszabályozási összefüggéseket tartalmazó rendszer. Ára: 18 000 Ft
5. Munkaügyi nyilvántartó rendszer, mely a dolgozói állomány kialakítását, karbantartását végzi. Ára: 40 000 Ft
6. Automatikus folyószámlakészítő feldolgozási rendszer. Ára: 39 000 Ft
7. Áruforgalmi folyamatok komplex értékesítési, információs feldolgozási rendszere. Ára: 39 000 Ft
8. Gyorsrendező program. Ára: 24 000 Ft
9. Teljes lemezmásoló program. Ára: 6 500 Ft

KERESKEDELMI SZERVEZÉSI INTÉZET
Bp. XIII., Dózsa Gy. út 150. Marketingosztály
Telefon: 202-650, 202-670